Mean Shift Segmentation

Final Report

Group 52
Shih-Chia Chen
Weiyi Jiang

2016.12.16

Literature Review

Mean shift is a non-parametric feature space analysis technique for seeking the maxima of the density function in a search space. It was originally introduced in 1975 by Fukunaga and Hostetler[1] and then generalized by Cheng in 1995[2]. At every iteration of this algorithm, the mean shift vector always points to the direction of the maximum increase in density. And different choices of the kernels in this algorithm will cause different results of mean shift vectors. The main strengths of the mean shift are it is capable of n-dimension feature spaces and it does not require any predefined parameters on given data set. Nowadays, mean shift algorithm has been used in a wide range of applications in image processing and computer vision such as image segmentation, tracking and smoothing.

In this project, we will implement the mean shift segmentation algorithm on test color images. Related techniques include discontinuity filtering, kernel choice, points clustering and image segmentation.

Introduction

In this project, our task is to implement the mean shift segmentation on a color image, this algorithm includes two major steps: discontinuity preserving filtering and mean shift image segmentation. We should locate the maxima of the density function of all points in the given image iteratively based on the idea of mean shift and then mark all points with homogeneous properties. Finally, we assign the new mean intensity value to points in image to obtain the segmentation result.

Approach

Algorithm:

The algoritms we use in this project are as follows:

1. Rearrange the original image to a feature vector matrix F containing both spatial and range information. F has six columns, the first three columns are the RGB values of the original image, the next two are (x, y) coordinates of this pixel in the image and the last column is a binary bit to mark whether this pixel has been marked.
2. While there exists pixel which has not been marked in F, we pick a random row from F as the seed.
3. Compute the Eucledian distance of all points in F with current seed and store the indices of all points whose distance from current seed is within threshold h. All such points are sample points in the range of a 5-dimensional ball whose center is at current seed.
4. Let h_s denote the size of window in spatial space, and h_r denote the size of window in range space. In this project, the spatial space is 2-demension and the range space is 3-demension. Then we compute the probability density for each point locating in the ball in (3) according to following equation:

$$K_{h_s,h_r}(x) = \frac{C}{h_s^2 h_r^2} k(\left\| \frac{x^s - x_i^s}{h_s^2} \right\|^2) k(\left\| \frac{x^r - x_i^r}{h_r^2} \right\|^2)$$

We use the Epannechikov Kernel in the calculation above:

$$k(x) = \begin{cases} C(1 - \|x\|^2), \|x\| \leq 1 \\ 0, otherwise \end{cases}$$

5. Use the probability density in (4) to compute the new mean:

$$y_{mean} = \frac{\sum_{i=1}^{n} x_i K_{h_s,h_r}(x)}{\sum_{i=1}^{n} K_{h_s,h_r}(x)}$$

Then we can get the mean shift vector $\overrightarrow{y_{mean}} - \overrightarrow{x_i}$

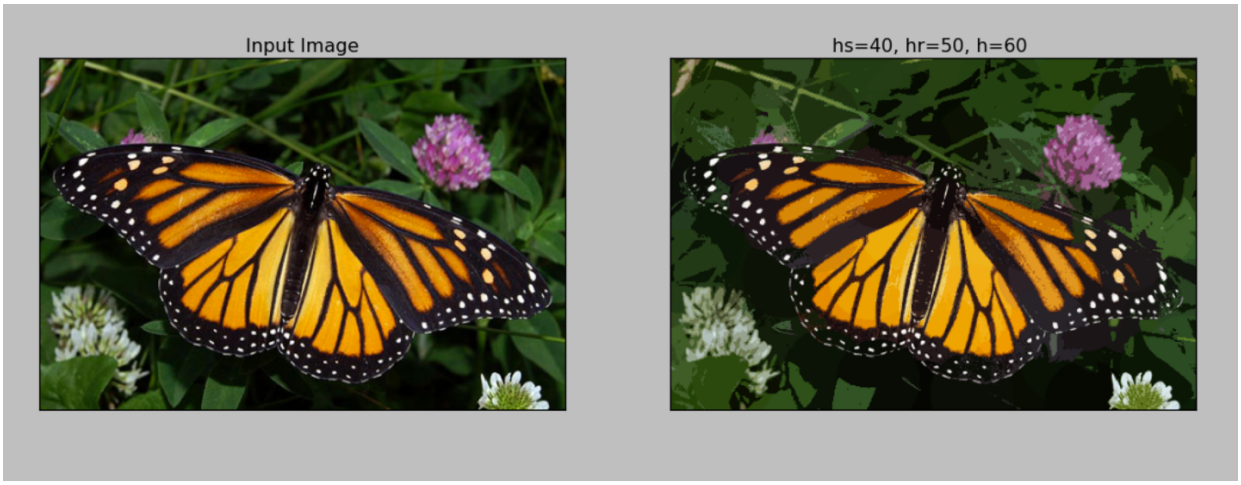
6. Check if the mean shift value is below a threshold ϵ, If it is below the threshold, assign the mean intensity values to all points selected in (3) to obtain the points cluster. Then removing all pixels that have been marked from the search space. Otherwise, choose a new random seed and repeat the iteration over again.

7. Continue all procedures until all pixels in given image are marked.

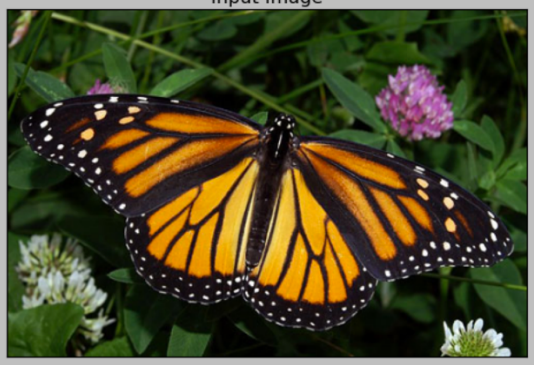
Implementation:
Our implementation is based on openCV-Python, numpy and matplotlib libraries. . We use openCV API to read color image and use matplotlib for displaying the results. In addition, we take advantage of the numpy library when we do algorithmic computation on the image matrix. However, we do not use any built-in functions related to operations about mean shift in openCV or numpy. We implement every step in our algorithm on our own.

Outcome and Deviations

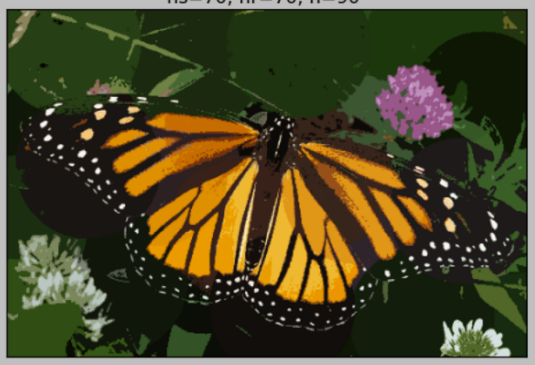
The results of the mean shift segmentation on given test images under different thresholds are as follows:



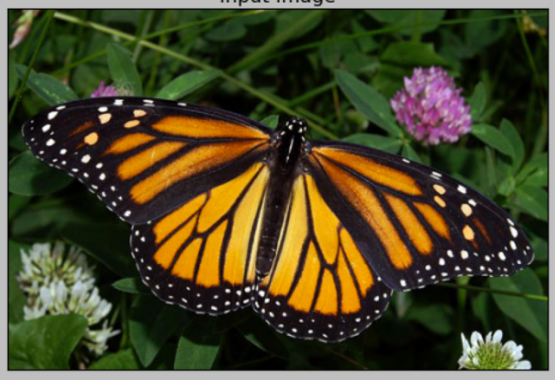
Input Image



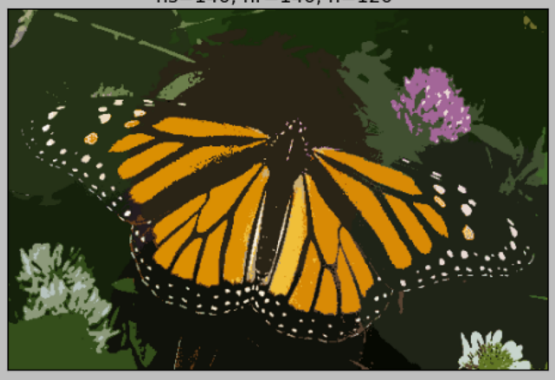
hs=70, hr=70, h=90



Input Image

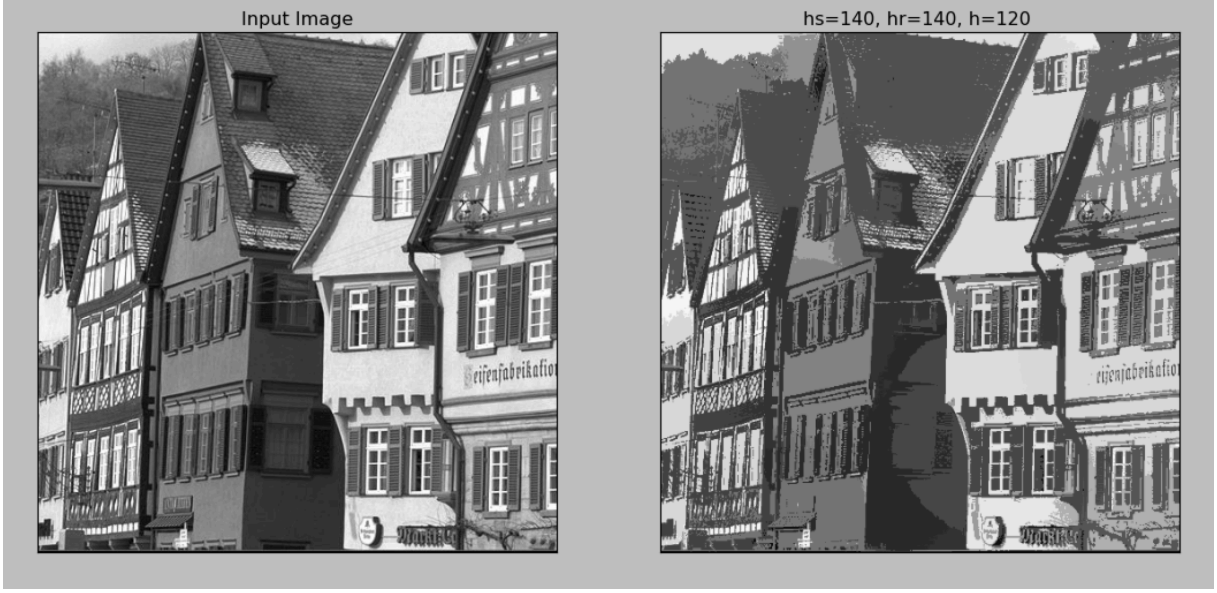


hs=140, hr=140, h=120

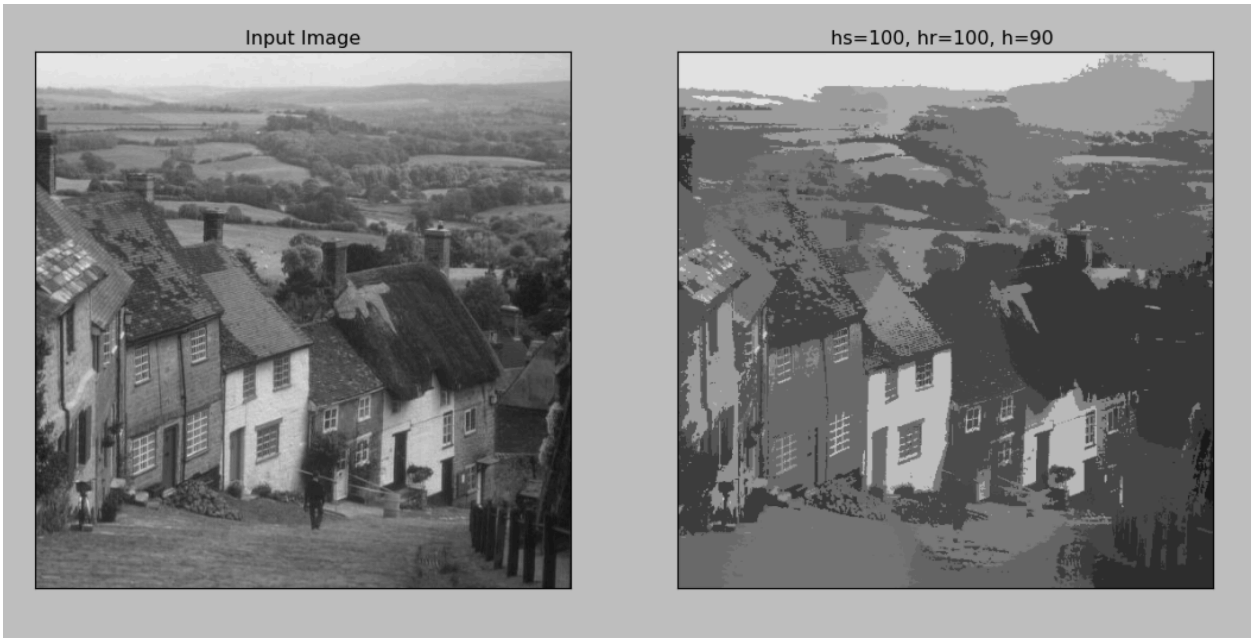


Input Image

hs=40, hr=50, h=60


Input Image
hs=140, hr=140, h=120


Input Image
hs=180, hr=180, h=140



If we run the algorithm with same parameters on the same image for several times, the output results may look a little bit different. This is because the selection of seed at each iteration is at random and the order of merging may be different every time we execute the code.

In conclusion, we find the most important parts of this algorithm are the choices of two range windows(h_r, h_s) and the computing of the mean shift vector. The choices of h_r and h_s are non-trivial and will make significant effects on the final results of segmentation, so we made many trials of different values of them to obtain results that look good. If we want to do better, using adaptive window sizes seems to be a

good idea. For calculation of the mean shift vector, due to the complexity of the mathematical equation, we should figure out how to translate the mathematical equation to the practical case in real world and write code to implement it.

Software and Program Development

Weiyi Jiang implemented the following parts of the program:
(1) Read image. Read the color image via imread function in openCV.
(2) Rearrange of the image. Suppose the size original image is m * n pixels, create a new matrix F with size (m * n) * 6. The first three columns are the RGB values of the pixel in original image, next two columns store the location of the same pixel, and finally, a binary mark bit to show whether this pixel has been marked. For example, if the RGB values of the first pixel is [50,20,30] in the image, the first row of F will be [50,20,30,0,0,0].
(3) Seed selection. Using the random number generator in Python to generate a random integer number in [0, number of rows in F] and check the bit mark of this row. If this pixel has been marked before, generate a new random number again. Otherwise, choose this row as the seed.
(4) Points selection. Compute the Eucledian distance of all pixels that have not been marked in F with the seed and determine a threshold h, then store the indices of all pixels that have distance within h.

Shih-Chia Chen implemented the following parts of the program:
(1) Computing new mean. Select the size of windows (h_s and h_r) and compute the probability density value of all points selected in previous step of a seed. Then calculating the new mean based on the probability density values.
(2) Computing mean shift vector. Using the new mean and seed to compute the mean shift vector. Determine a threshold ϵ , if the length of the mean shift vector is within ϵ, then go to the next step. Otherwise, select a new seed and repeat all procedures again.
(3) Assign new values to the pixels. If the mean shift vector in (2) is accepted, then assign the mean intensity values to all points in the original image, and set the bit of all these pixels to 1.
(4) Results. After all pixels are marked, display the result image via subplot and imshow function of matplotlib.

During the implementation, we find we should take advantage of the build-in functions about vector operations and matrix computation to reduce the running

time of our program. In our first version of implementation, we wrote lots of loops to do such computation and it always took several tens of minutes to get a result. Then we modified our implementation and started to use the build-in functions in numpy to do these calculations. After doing that, we could get the results in a few minutes. These functions really help us save a lot of time.

Summary

By implementing this project, we have a deeper understanding of the methodology of the mean shift and the physics in it. We also gain much experience in image processing and scientific computing via Python.

From the course during this semester, we have learned the basic knowledge about image processing, both in concepts, and the mathematical operations that can be operated on an image. Moreover, we not only learn the concepts through the lecture, but also have a chance to implement some techniques and operations in programming homework and the final project. In conclusion, we master the basic principles in digital representation of image, basic techniques for image manipulation and programming skills to implement image processing algorithms after taking this course.

Reference List

[1] K. Fukunaga and L.D. Hostetler, "The estimation of the gradient of **a** density function, with applications in pattern recognition," ***IEEE*** *Truns. Infi,rmution Theory,* vol. 21, pp. 32-40, 1975.

[2] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Tans. Pattern Analysis and Machine Intelligence*, Vol. 17, No. 8, pp. 790-799, August 1995.

[3] "Mean Shift" from Wikipedia: https://en.wikipedia.org/wiki/Mean_shift.