The Design and Implementation of SLIC Superpixel Algorithm

Kai Wang¹,a, Jing Di²,b, Hongwei Zhao³,c

¹,³College of Computer Technology, Jilin University, Changchun 130012, China
²Network Center, Jilin Normal University, Changchun 130012, China
⁰email: wangkai19930418@163.com, ¹email: dijing@cncnc.edu.cn
³Corresponding Author: zhaohw@jlu.edu.cn

Keywords: Superpixel, SLIC Algorithm, Image Segmentation

Abstract. The concept of superpixel firstly appeared in the research of image segmentation. Superpixel algorithms are designed to segment an image into a series of blocks, all these blocks are comprised of a number of pixels. By composing a superpixel network instead of making a pixel network, we can deal with every superpixel as a separate entity. By this means, we reduce the time complexity of the original algorithms which deal with pixels one by one. And always we could get better results via improved algorithms aimed at superpixels. The essence of superpixel segmentation is an abstraction of basic information in an image. One of the best superpixel segmentation algorithm is SLIC algorithm, we will introduce it and implement it by C programming language.

1. Introduction

Ren [1] firstly came up with the concept of superpixel in the research of image segmentation. By the classification and abstraction of image information, superpixel algorithms divide an image into a number of image blocks named superpixels. Compared with pixels, these superpixels seems more separate and complete, and they have clearer semantics. All these help us with dealing with images effectively.

According to the differences of algorithm ideas, superpixel algorithms can be classified into two categories: Graph-based algorithms and Gradient Descent algorithms:

(1) Graph-based algorithms: Graph-based algorithm [2], Ncut algorithms [3], Superpixel Lattice algorithm [4], Entropy-based algorithm [5].

(2) Gradient Descent algorithms (mainly based on clustering): Watershed algorithm [6], Mean Shift [7], Quick Shift [8], Tuborpixels [9], SLIC algorithm [10].

One of the most effective algorithms is SLIC algorithm. The idea of this method is simple to comprehend, easy to implement and convenient to use in practical ways. This algorithm computes indicators which are relevant to the superpixel segmentation by easy-access features such as color features and distance features in an image. Superpixels obtained by this method are more complete and consistent than other algorithms. Moreover, the sizes of superpixels are approximately the same, and the shapes of these superpixels seems more regular.

2. SLIC Algorithm

SLIC algorithms is proposed by Achanta [10] in 2010. Before the appearance of this method, there are seldom effective superpixel algorithms which can segment an image into regular, complete and consistent superpixels with few computations. The advantages of SLIC algorithm are simplicity,
effectiveness and speediness. By one specific parameter, SLIC algorithm can take control of the number of segmented superpixels in the end.

In this method, the superpixels are obtained by pixels clustering based on the similarity in color features and the distances between pixels. First, we will transform an image from RGB color space to CIELAB color space and get the L,A,B values of every pixel. By combining the LAB values with XY coordinates of one pixel, we can get five numbers to identify the pixel, that’s 5-d LABXY feature vector. Then, we need to compose a novel and effective criterion to measure the similarity between pixels to complete the clustering of image pixels. The concrete procedures of this algorithm is as follows:

(1) Initializing clustering seeds

Supposing the image is composed of N pixels, and we plan to divide it into K superpixels with almost the same sizes, then every superpixel will be made up of N/K pixels, and the distance between adjacent clustering seeds is approximately S=sqrt(N/K). At first, we should separate the image into S*S square pixel blocks and initial the square centers as the clustering centers.

To improve the computational speed of the algorithm, we suppose that the similar pixels for a clustering center just appear in its neighborhood. So every clustering center will only search for its similar pixels in the 2S*2S square area around it instead of searching in the whole image.

(2) The measure of similarity

For the pixels near to one specific pixel, the Euclid distance plays an important role in the measurement for similarities between them. However, as the distances become larger, the effect of the Euclid distance will recede.

For every pixel, we compare it with its nearby clustering seeds by computing the similarity values via Eq.1 below. The smaller the value is, the more similar these two pixels will be. And then we shall assign every pixel the label number of its most similar clustering seeds.

The formula for computing similarity values is:

\[ D_{lab} = \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2} \]
\[ D_{xy} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \]
\[ D_s = D_{lab} + \frac{m}{S} D_{xy} \]  

In the formula: \( D_{lab} \) represents the distance in LAB color space between two pixels, \( D_{xy} \) represents the distance in XY coordinate space, and \( D_s \) represents the similarity value between these two pixels. The larger \( D_s \) is, the more similar these two pixel will be. In the formula for computing \( D_s \), \( S \) means the distance between adjacent seeds, it’s \( S=\text{sqrt}(N/K) \) as shown below, \( m \) is a balance item in the formula, it can adjust the insistence of the ultimate results. When \( m \) has a large value, the \( D_{xy} \) will show great effect in the computation of \( D_s \), otherwise, \( D_{lab} \) will be more important here. For the parameter \( m \), we recommend it to be within [1,20]. In practical, we always set \( m=10 \) to balance the weight between \( D_{lab} \) and \( D_s \) so as to have better results.

By iterating this process to a convergence, every pixel will pertain to a certain clustering center in the end. And every clustering seed stands for a certain area of pixel block named superpixel.
Next, we are going to calculate new centers for these superpixels. The LABXY values for all new centers can be obtained by computing the average LABXY values of all pixels in this superpixel area.

After these operations, sometimes we can also find some very small superpixels in the image, they are near to some large superpixels but don’t belong to them. To make the usage of superpixel algorithm more convenient, we shall merge these small superpixels with their nearest large superpixels.

In the figures below, the certain image has been approximately segmented into 200, 300 and 400 superpixels.

![Figure 1](image1.jpg)

Figure 1. From left to right, the image is segmented into 200, 300 and 400 superpixels.

Here are some sample pictures below to show the effect of SLIC algorithm:

![Figure 2](image2.jpg)

Figure 2. Source Images.

![Figure 3](image3.jpg)

Figure 3. The ultimate images segmented by SLIC algorithm.

3. The implementation of SLIC algorithm

The implementation of SLIC algorithm given by Achanta\cite{10} is programmed by Matlab. Matlab is a convenient tool for implementing an algorithm. However, as we all know, Matlab has the disadvantage that it shows very slow speed in arithmetic operations, especially when there are many loops in the program. So we try to take advantage of C programming language to implement the SLIC algorithm, in order to enhance the executive speed of our program.

The main functions we implemented by C language is shown as follows. The former three are subprograms always called by the function (4).

(1) Declaration: void rgb2lab(int* red, int* green, int* blue, int size, double* lvector, double* avector, double* bvector)
Function: Convert the resource images from RGB color space to LAB color space.

(2) Declaration: void getSeeds(int step, int width, int height, int* seedIndices, int* numofseeds)

Function: Initial the position of clustering seeds.

(3) Declaration: void ImproveSuperpixelConsistence(int* labels, int width, int height, int NumofSuperpixels, int* nlabels, int* finalNumberOfLabels)

Function: Deal with all the small superpixels after the first segmentation.

(4) Declaration: void SLIC(double* lvector, double* avector, double* bvector, double* kseedsl, double* kseedsa, double* kseedsb, double* kseedsx, double* kseedsy, int width, int height, int numofseeds, int* klabels, int step, double compactness)

Function: Complete the SLIC superpixel segmentation algorithm with using the former three functions.

4. Summary

In this article, we introduce an excellent superpixel algorithm named SLIC algorithm. SLIC algorithm is simple to implement and effective to use. It segments an image into a number of superpixels by taking advantage of color features and distance features of the image. We implement this method by C programming language to improve the executive speed. In the end, it shows excellent time complexity and all superpixels have almost the uniform sizes and regular shapes.

5. Acknowledgment

The corresponding author is Hongwei Zhao. The authors are grateful to the anonymous reviewers for their insightful comments which have certainly improved this paper. This work is supported by Natural Science Foundation of Jilin Province (20140101184JC) and Special funds for industrial innovation in Jilin Province (2016C035).

References


