UDC 004.932.2

R. Vyshnevskyi

RESEARCH OF GRAPH–BASED IMAGE SEGMENTATION ALGORITHM

Abstract. *This article is dedicated to one of the basic problems in computer vision – image segmentation. The main methods of image segmentation, a process of division of an image into different segments which contain pixels of similar color, are considered. A graph-based segmentation algorithm, which is based on Kruskal's algorithm is implemented in C++ language.*

Keywords: *segmentation, image, computer vision, spanning tree, Kruskal's algorithm, graph, threshold function, methods of segmentation, intensity, Quicksort algorithm*

Р. А. Вишневский

ИССЛЕДОВАНИЕ АЛГОРИТМА СЕГМЕНТАЦИИ ИЗОБРАЖЕНИЙ НА ОСНОВЕ ГРАФОВ

Аннотация. *Статья посвящена одному из основных вопросов компьютерного зрения – сегментации изображений. Рассмотрены основные подходы к сегментации изображения – процессу разделения изображенния на сегменты, которые содержат в себе пиксели подобного цвета. Алгоритм, базирующийся на алгоритме Крускала, реализован на языке C++.*

Ключевые слова: *сегментация, изображение, компьютерное зрение, остовное дерево, алгоритм Крускала, граф, пороговая функция, методы сегментации, интенсивность, быстрая сортировка*

Р. А. Вишневський

ДОСЛІДЖЕННЯ АЛГОРИТМА СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ ГРАФІВ

Анотація. *Стаття присвячена одному з основних питань комп'ютерного зору – сегментації зображень. Розглянуто основні підходи до сегментації зображення – процесу розділення зображення на сегменти, які містять у собі пікселі схожого кольору. Алгоритм, що базується на алгоритмі Крускала, реалізовано мовою C++.*

Ключові слова: *сегментація, зображення, комп'ютерний зір, кістякове дерево, алгоритм Крускала, граф, порогова функція, методи сегментації, інтенсивність, швидке сортування*

Introduction

Computer vision is one of the areas, that are most in demand at this stage of development of global digital computer technologies. Computer vision technologies are used in industry, process automation, robotics, medicine and military purposes [1].

The basic problems of computer vision are image segmentation and edge detection, which are used in larger problems such as scene recognition and object detection. The image segmentation problem is one of the oldest and hardest in computer vision. During last years we can see a big progress in this area [2, 3].

Segmentation is a process of division of an image into segments (sets of pixels). Its aim is to simplify the image for further analysis. Segmentation is usually used to highlight objects and boundaries on the image [4, 5].

Methods of segmentation

The main methods of segmentation are: split-and-merge methods, region-growing methods, Markov field modeling, edge detection and graph partitioning methods [2, 4].

The main idea of region-growing methods is [2]: first we choose seeds (some pixels of the image) and then we unite adjacent pixels to the seeds if they satisfy some criterion. The process ends when there is no pixels that can be united.

Split-and-merge methods consist of 2 main steps [2]: split and merge. Split is applied to the image until we get oversegmentated image. After that adjacent similar segments are united until we get division on uniform areas of maximal size.

Markov field model is based on assumption that color of every pixel depends on colors of a set of adjacent pixels. This method is hard to implement but is effective when you need to take into account the texture of an image [2].

Edge detection methods work good for grayscale images. The image is considered as a function of two variables. The edges of areas are the maximums of the gradients of this function. Different filters are used to find the gradients.

The main lack of these methods is that they are unstable to noise [2].

In this work a method, based on graph theory [6, 7] is studied. Graph partitioning methods are one of the areas in image segmentation which are actively developing. The main idea of methods of this area is [2]: an image is presented in a form of a weighted graph with vertices in pixels of the image. The weight of an edge represents the similarity between pixels (the distance between pixels in a certain metrics).

These methods are easy to implement. The main problem of graph partitioning methods is a large memory usage because the majority of methods keep in memory a matrix of distances between pixels [2]. But their main advantage is that they work fast.

The results achieved by different algorithms can be found in [2] and [8].

Formulation of the problem

The input image must be divided into areas with similar colors. As an input we have an image and 2 parameters:

1) segmentation parameter which determines the amount of areas;

2) number of iterations for blur algorithm, which allows to get better result (blur removes noise and artifacts).

As an output the user gets a segmented image, divided into areas of pixels with similar colors. The result varies according to input parameters.

The aim of this work is to analyze existing researches of the segmentation algorithm concerning its speed of operation.

Efficient graph-based image segmentation

In this work properties of efficient graph-based image segmentation algorithm, proposed by P. Felzenszwalb, are considered. In particular, its speed of operation is estimated.

The segmentation algorithm is based on Kruskal's algorithm, which finds a minimum spanning tree (MST) for a graph [9].

According to [10] every pixel of an image is represented by a vertex in a graph. The weight of an edge which connects adjacent vertices is:

$$w(v_i, v_j) = | I(p_i) - I(p_j) | ,$$

Where $I(p_i)$ – intensity (brightness) of pixel p_i.

In the course of Kruskal's algorithm, on the intermediate step we will have several distinct segments (sets of pixels) with a minimal sum of weights of edges inside: segments will be united by edges with minimal weights (with minimal difference of intensity between adjacent pixels). That means that adjacent pixels inside a segment will have similar colors but it will be lower than the certain maximum value of edge (difference of intensity).

We will examine the minimal edge on current step. For two pixels which are adjacent vertices of this edge we will determine if they are from the same segment. If they are from the same segment, we continue the algorithm; otherwise, we must determine if the segments represent the parts of the same object on the image and must be united or if their intensities differ too much.

To determine difference between segments we will associate a certain value with every segment which is already built – maximal difference of intensity inside it (the edge with maximal weight in MST of the segment) [10]:

$$Int(C) = \max_{e \in MST(C)} w(e) . \quad (1)$$

We have a rough decision rule for segments C_1, C_2 according to which we decide if the segments must be united:

$$D(C_1, C_2) = \begin{cases} true, Diff(C_1, C_2) > MInt(C_1, C_2) \\ false, otherwise \end{cases} . \quad (2)$$

$D(C_1, C_2)$ is a property which determines if we need to separate the segments.

Current edge of minimal weight which unites two segments:

$$Diff(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} (w(v_i, v_j)) . \quad (3)$$

The minimum (of the maximum) difference of intensity inside one of the considered segments:

$$MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2)) . \quad (4)$$

To make pixels unite more than bigger segment which are already built and for which the correctness of division is more important we have added a threshold function $\tau(C)$.

$$\tau(C) = k / | C |,$$

where k – parameter of segmentation, which is input by a user; $|C|$ – power of considered segment (number of pixels in it in current moment of time).

The correction is reduced step by step as the segment grows ($|C|$ becomes bigger).

According to the value of parameter k, the number of segments will vary. The bigger the value of k, the less segments will the output image consist of. This allows to adjust the segmentation degree.

We can choose another function as $\tau(C)$ that will consider the specificity of input images: form of segments, certain tints of colors etc.

This means that to unite segments the difference of intensity on their boundary must be less than maximal difference inside each of the segments:

$$Diff(C_1, C_2) < MInt(C_1, C_2). \quad (5)$$

There are two ways to build a graph for an image [10]:

- 4-connectivity: every pixel is connected to adjacent top, down, left, right ones. That means that the number of edges is minimal;
- 8-connectivity: additionally to previous variant every pixel is connected to his diagonal neighbors. That way we have more edges in the graph and the algorithm runs a little slower. But the resulting segmentation is more qualitative because we consider more connections than in previous variant.

The algorithm should produce more qualitative results on 8-connected graph but 4-connected graph produces acceptable results and the algorithm works faster.

As a distance between pixels we can choose a difference between colors:

$$dist(p_i, p_j) = \sqrt{(r_i - r_j)^2 + (g_i - g_j)^2 + (b_i - b_j)^2}. \quad (6)$$

The main steps of algorithm are:

Data: input image.

Result: segmented image apply Gaussian blur;

– build edges (their weights are calculated with formula 6);

– sort edges in ascending order;

– unite similar segments;

– output image is built.

Computational complexity

More closely this problem is studied by [8].

We agree with specialists [10] that the algorithm is fast but not very qualitative.

The most of time is spend on sorting of all the edges. To sort the edges, we have chosen Quicksort algorithm. Its complexity is $O(n \log n)$ where n – number of edges in graph [8].

The algorithm was slightly improved. Instead of keeping in memory edges (p_i, p_j) and (p_j, p_i) only one of them is kept. Thus, number of edges if reduced from 4*m*n-2*m-2*n to 2*m*n-m-n, where m and n are width and height of input image.

The complexity of segments uniting is O(n). For Gaussian blur it is also O(n).

Memory complexity is O(m*n), because we need to keep in memory m*n pixels and N edges.

Algorithm testing was done on Berkeley image segmentation database. To compare qualitative characteristics of the algorithm with others, results from [8] were used.

Conclusion

Based on previous theoretical and practical development in image segmentation we studied an algorithm which uses graph theory for image segmentation [8, 10] and a program which divides an image into areas with similar colors was made.

We discovered that this algorithm is fast but does not give very precise results. For better quality [10] proposes to connect each vertex with adjacent ones in radius R. On the other side, this will make the algorithm work slower and will increase memory costs.

Algorithm can be used in different areas, such as medicine (to discover pathologies, injured tissues, in microsurgery etc.), machine vision (segmentation is a part of more complex pattern recognition algorithms).

Therefore, recognizing existing solutions and research of the segmentation algorithm, we can assert that the proposed improvements allow increasing the quality of segmentation, but performance of the algorithm becomes worse. The problem of improvement

of image segmentation algorithms is actual and needs to be studied further.

References

1. Vezhnevets V., and Konushin A. Vvedenie v komp'yuternoe zrenie [Introduction to Computer Vision], (2006), *MGU VMK, Graphics & Media Lab* (In Russian).
2. Vejnevets A., and Barinova O. Metody segmentatsii izobrazhenii: avtomaticheskaya segmentatsiya [Image Segmentation Methods: Automatic Segmentation] (In Russian), url: http://cgm.computergraphics.ru/content/view/147.
3. Fulkerson B., and Soatto S. Really Quick Shift: Image Segmentation on a GPU, 2010, *PhD thesis, Department of Computer Science, University of California,* Los Angeles (In English).
4. Porshnev S., and Levashkina A. Universal'naya klassifikatsiya algoritmov segmentatsii izobrazhenii [Universal Classification of Image Segmentation Algorithms], (2008), *Journal of Scientific Publications of Postgraduates and PhDs*, No. 4, Vol. 14, 11 p. (In Russian).
5. Fisenko V., and Fisenko T. Komp'yuternaya obrabotka i raspoznavanie izobrazhenii [Computer Processing and Image Recognition], (2008), *SPbGUITMO*, pp. 155 – 164 (In Russian).
6. Diestel R. Graph Theory, (2005), *NY: Springer-Verlag* (In English).
7. Bondy J., and Murty U. Graph Theory 2008. *NY: Springer* (In English).
8. Pantofaru C., and Hebert M.A Comparison of Image Segmentation Algorithms, (2005), *PhD thesis, The Robotics Institute, Carnegie Mellon University,* Pittsburgh, Pennsylvania (In English).
9. Kruskal J. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *AMS*, Vol. 7. Proc., 1956 (In English).
10. Felzenszwalb P. Efficient Graph-based Image Segmentation, (2004), *International Journal of Computer Vision*, Vol. 59 (In English).

Received 03.03.2014

Vyshnevskyi
Rostyslav Anatoliiovych,
Student of Faculty of Cybernetics Taras Shevchenko National University of Kyiv,
02140, Kyiv, Grishka st., 10, ap.14,
tel.:096-799-86-11.
E-mail:
mr.jumpy92@gmail.com.