Pattern Recognition Letters 87 (2017) 155–162

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Sequential image segmentation based on minimum spanning tree representation☆

Ali Saglam*, Nurdan Akhan Baykan

Department of Computer Engineering, Faculty of Engineering, Selcuk University, Konya 42130, Turkey

ARTICLE INFO

Article history:
Available online 16 June 2016

Keywords:
Segmentation
Clustering
Graph
Minimum spanning tree
Prim

ABSTRACT

Image segmentation is a very important stage in various image processing applications. Segmentation of pixels of an image and clustering of data are closely related to each other. For many graph-based data-clustering methods and many graph-based image-segmentation methods, minimum spanning tree (MST)-based approaches play a crucial role because of their ease of operation and low computational complexity. In this paper, we improve a successful data-clustering algorithm that uses Prim's sequential representation of MST, for the purpose of image segmentation. The algorithm runs by scanning the complete MST structure of the entire image, such that it finds, and then cuts, inconsistent edges among a constantly changing juxtaposed edge string whose elements are obtained from the MST at a specific length. In our method, the length of the string not only determines the edges to compare, but also helps to remove the small, undesired cluster particles. We also develop a new predicate for the cutting criterion. The criterion takes into account several local and global features that differ from image to image. We test our algorithm on a database that consists of real images. The results show that the proposed method can compete with the most popular image segmentation algorithms in terms of low execution time.

1. Introduction

Image segmentation is an image clustering problem; it is one of the most important and difficult computer vision and pattern recognition problems. This problem includes the process of dividing an image into homogeneous regions or segments such that the homogeneity can be measured based on the similarity of the properties of the image, such as the gray level, color, or texture [1–3]. Using the texture features of the image has a high computation complexity [4]. On the other hand, grayscale images inherently supply less information than color images [5]. Therefore, in many computer vision and pattern recognition applications, color image processing is more popular because of its practicality and accuracy [6]. In color image processing, appropriate color spaces and metrics can be used to measure similarities [5]. To perform the segmentation process, certain algorithms use global information extracted from the entire image, whereas other only use local information obtained from within the image. Many methods that use global information take a long time, whereas those that use only local information take a short time, but are usually misdirected by noise [7].

Clustering is an unsupervised process that categorizes data into groups, as in the image segmentation process [8]. According to Wertheimer's gestalt theory [9], image segmentation has a close relationship with data clustering as a perceptual process [10]. Therefore, in the following sections, both (indeed, simultaneous) data clustering and image segmentation are of interest.

In graph-based approaches to image segmentation, an image is mapped onto a plane using graph theory tools [11]. Graph theory helps to obtain information about the properties of the image. Graph-based image segmentation groups and organizes feature information and spatial information [3]. In addition to feature similarity, graph-based clustering approaches also take into account the connectivity and structural similarity of vertices (or nodes), which indicate pixels in the image [12,13]. After mapping the image onto a graph, the segmentation process is performed in a discrete connectivity space; thus, graph-based segmentation or clustering methods do not require discretization and do not produce any discretization error [10]. Because of the effective and powerful data representation (in addition to other advantages), the graph-based approach has been employed in many popular image segmentation and data clustering methods [7,13–18].

☆ This paper has been recommended for acceptance by Cheng-Lin Liu.

* Corresponding author. Tel.: +90 332 223 1972.
E-mail addresses: alisaglam@selcuk.edu.tr, alisaglam666@gmail.com (A. Saglam).

http://dx.doi.org/10.1016/j.patrec.2016.06.001
0167-8655/

In graph-based color image processing, pixels can be assumed to be vertices, and images with a size of A pixels in height and B pixels in width can be assumed to be an $A \times B$ vertex array. Each pixel has a color value vector in a color space (e.g., RGB). An edge is a link that connects the two vertices; in a weighted graph, each edge has a quantitative cost, which can be represented (for example) as the color difference between two adjacent vertices in the image graphs, such that it can be set using a distance measurement such as the Euclidean distance in the color space [5,7]. In graph-based image processing, vertices are connected to each other via a certain number of neighbors, such as in 8-neighboring or 8-connected vertices [2,7].

A spanning tree is a weighted, undirected, and acyclic subgraph. If there is only one path between all the pairs of vertices in a graph (i.e., there are no cycles or loops in the graph), the graph is acyclic. If each edge has an assigned orientation to another edge in a graph, the graph is a directed graph; otherwise, the graph is an undirected graph. A minimum spanning tree (MST) or a shortest spanning tree (SST) is a spanning tree with the least total weight of all edges among the all possible spanning trees [2,11,19]. The most popular MST extracting algorithms are Boruvka's algorithm [20], Kruskal's algorithm [21], and Prim's algorithm [22]. Boruvka's algorithm forms the MST by finding the nearest vertex for each vertex, and then, the nearest vertex for each tree, and merges them hierarchically such that no cycle is formed. Kruskal's algorithm forms the MST by listing all of the edges in ascending order and adding to the MST at the lowest priority, without generating any cycles. Prim's algorithm forms the MST by finding the nearest edge to the tree that has been formed by beginning from an arbitrary vertex and adding to the tree, such that the edge to be added will not form a cycle.

Many graph-based clustering algorithms and graph-based image-segmentation algorithms have widely used MST structures and algorithms because of their high performance, ease of retention of data and clusters, and ease of operation. The most popular MST-based clustering and segmentation algorithms will be presented in the next chapter. Our method is inspired by a clustering algorithm proposed by Wang et al. that is a recent MST-based clustering algorithm [23]. To operate the algorithm, first, Prim's algorithm is run to extract the MST that spans all of the data. Prim's algorithm adds one edge to the MST structure being formed at each cycle and into a one-dimensional edge list at the same time. After obtaining the MST, the clustering algorithm composes an edge list from the added edges in the order of their addition. Finally, an edge string consisting of edges from the list, the length of which is determined previously by users, is taken forward through the list; during this process, the edge at the exact center of the string is cut if it is the maximum weighted edge among the edges in the string. Although the algorithm gives quite good results for data clustering, when it is directly applied to image segmentation, it fails, unlike other segmentation algorithms. Adapting this method to different images using only the length of the string is very difficult, because, there is no adaptive threshold value that varies according to the characteristics of the given image. The time complexity of the algorithm is in a good quality [23].

In the proposed algorithm, we present a new predicate as a cutting criterion. Instead of the entire frame, we separately consider both sides of the center of the frame and use a threshold value for adaptation. The threshold value can be manually determined by the user to control the segmentation or automatically determined using the method presented in this paper. Moreover, we reduce noise segments using the given parameter for the length of the edge string. In this paper, we apply the algorithm to real images from the Berkeley Image Segmentation Dataset [24]. Our experimental results show that our method is able to compete with the most popular image segmentation algorithms.

2. Releated works

Image segmentation and clustering is a very challenging problem. No method among the clustering and segmentation methods achieves complete success. Several perform well, but take a very long time. Others are fast, but cannot prevent noise. Moreover, no single successful method is suitable for all data. In this section, we briefly mention some of the studies that are relevant to data clustering and image segmentation, i.e., those that use the MST structure. These studies have a long history.

In 1969, Gower and Rose [25] presented two iterative algorithms which are both called MST-based single linkage clustering. These methods are more practical than traditional clustering methods up to that time, because they use an algorithm that finds the MST, and thus, the results are obtained in a short computing time. Both methods perform clustering using a threshold value on the MST of data. Zhan [13] proposed an MST-based method that is based on the global properties of data. This method calculates a different threshold value for each edge using the standard deviations and means of the connected vertices at the ends of the edge, and a constant value. If the weight of the edge is larger than the threshold, the edge is cut.

Morris et al. [2] proposed a recursive MST-based image segmentation method called recursive SST segmentation. This method uses Kruskal's algorithm. Initially, each vertex defines a region. The two adjacent regions at the end of the smallest edge are merged. Then, the merged regions are combined into the same region, and the values of the vertices in the new region are altered based on the mean of the new region. Thus, new weights of the edges are updated according to the new region value. There should be one shortest edge between each adjacent region. In each cycle, the processes are repeated until the desired cluster number is reached.

Xu and Uberbacher [7] proposed an MST-based image segmentation method on greyscale images using Euclidean distances. This method tries to create homogenous regions by minimizing the sum of variations of all regions. To achieve this goal, each subtree should have greater than or equal to a specified number of vertices. Additionally, neighboring subtrees should be noticeably different from each other.

Xu et al. [17] developed three clustering algorithms using Euclidean distance. The first algorithm simply partitions the MST of the data by cutting the $k-1$ longest edge for k cluster. The second algorithm iteratively partitions the MST by decreasing the total difference between each vertex and the mean value of the cluster in which the vertices exist; the method uses an objective function to perform this task. The third algorithm finds the globally optimal solution to the clustering problem. This method selects k representative vertices based on minimization of the total weight between the each vertex and its closest representative vertices. Similarly, the method also uses a slightly differently objective function from previous one. Olman et al. [26] present a one-dimensional clustering approach that is useful for our method. According to the approach, a clustering problem is transformed to a string-partitioning problem using Prim's sequential MST representation.

Felzenszwalb and Huttenlocher [15] define a predicate for measuring the evidence between neighboring regions. The method uses not only local information, but also global information regarding the regions, such as cluster size and longest edge for each cluster. However, this method generates many small segment particles in the segmentation result. This method requires another process to eliminate the particles. Haxhimusa and Kropatsch [18] present a hierarchical partitioning method using Boruvka's algorithm and benefiting from the homogeneity criterion of [15].

Grygorash et al. [27] proposed two methods, one takes into account the given value for the number of clusters, and the other takes into account a threshold value. These methods are similar to

Xu and Uberbacher's method, but they minimize the variation of each cluster one-by-one, rather than by their sum.

Zhong et al. [28] employed two rounds of MST. The method classified cluster problems into two groups as separated into the clustering problem and the touching-clustering problem (in the literature, these procedures are known as the split problem and the merge problem, respectively). Therefore, they proposed two algorithms for these problems that are run repeatedly on two different MST structures created from the same graph. Later, they proposed a new split and merge clustering method [29]. The problem uses an MST-based graph for initial prototypes and splits the graph using a K-means algorithm [8], which is a common in traditional clustering algorithms because of its ease of use. Next, the sub-graphs are filtered and then merged by considering neighboring edges.

Wang et al. [23] assumed a sequential clustering algorithm; our method takes inspiration from this technique. The working principle of this method was mentioned in Section 1. In addition, they also proposed a k-partition method. The edge array, which is used for finding inconsistent edges from the sequential edge list formed during Prim's algorithm, is run for all possible string lengths; and a set of inconsistent edge queues is created. A queue is selected among them based on the given cluster number k.

3. Preliminaries

3.1. Graph clustering representation of an image

Let $G = (V, E)$ be an undirected, weighted graph with vertices $V = \{v_1, v_2, \ldots, v_n\}$ and edges $E = \{e_1, e_2, \ldots, e_m\}$. Each vertex v_p refers to a pixel in the given image data, and each edge $e_q = (u, v)$ refers to a link that connects the vertex u to v, which is an adjacent vertex, where $p = 1 : n$ and $q = 1 : m$. In this paper, we connect the vertices to one another based on the 8-connected neighborhood. Each edge has a weight $w(e_q)$ or $w(u, v)$. To measure the weight of each edge, in this paper, we use a Euclidean distance measurement within the RGB color space using the color feature of each pixel.

In graph-based image segmentation, the image segmentation process is reduced to the problem of partitioning the graph mapped from the image [2,7,15]. On the basis of that, each subgraph C of G refers to a segment of the image. Similarly, in the MST-based clustering and image segmentation methods, the graph partitioning problem is also reduced to a tree partitioning problem. In this method, each subtree represents a sub-graph [2,7,15,23,29].

Let an MST $T = (V, E_T)$ be a minimum weighted subtree of the graph G that covers all vertices V of G. If $|V|$, which refers the number of vertices of T, is n, $|E_T|$ refers to the number of edges of T, i.e., $n - 1$ [21]. If an edge e_q is cut, where $E_T \neq \varnothing$, $e_q \in E_T$, T would be separated into two trees, i.e., $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$. The tree partitioning process would be concluded as follows: $V = V_1 \cup V_2$, $E_T = E_1 \cup E_2 \cup e_q$, $e_q \notin E_1$, $e_q \notin E_2$, $V_1 \cap V_2 = \varnothing$, $E_1 \cap E_2 = \varnothing$. Thus, each tree T_1 and T_2 is a spanning tree of the subgraphs C_1 and C_2, respectively. Therefore, each tree represents a segment of the image [7].

3.2. Definitions of some clustering problems

In this paper, several common clustering problems are considered. These problems, and some factors that affect their solutions, are defined as follows:

Definition 1. $Int(T)$ is an *internal difference* [15] that refers to the largest edge weight in the tree T.

Definition 2. A cutting process results after deleting an *inconsistent edge* (e.g. e_1, e_2, e_3, and e_4 in Fig. 1) such that it satisfies an inconsistency criterion [13,27].

In MST-based clustering approaches, the segmentation process relies on the problem of which edge on the MST to cut. In this case, the bases of the principles of Gestalt theory are added to determine the edges to be cut. According to these principles, the cut edges should be considerably different from nearby edges. In the literature, this condition is referred to as inconsistency [13]. To decide that an edge causes this inconsistent situation, an inconsistency criterion is applied to the edge.

Definition 3. An *explicit edge* is an inconsistent edge such that it separates a tree to two subtrees that are similar in density; its weight is larger than the internal differences of both of the subtrees.

Many clustering methods exploit the similarity of elements to separate them into clusters. We assume that a cluster has a high density if elements in the cluster are very similar to each other. If the similarity is lower, the density of the cluster is lower. According to this relation, the density of a tree structure that consists of low weighted edges is high and vice versa [13].

In Fig. 1(a), we assume that the density of T_1 and the density of T_2 are similar, $w(e_1) > Int(T_1)$, and $w(e_1) > Int(T_2)$. Thus, e_1 is called an explicit edge.

Definition 4. A *transition edge* is an inconsistent edge such that it separates a tree into two subtrees that are different in density; its weight is larger than the internal difference of one of the subtrees, whereas the weight is smaller than the internal difference of the other subtree.

In Fig. 1(b), we assume that the density of T_3 and the density of T_4 are different, $w(e_2) > Int(T_3)$, and $w(e_2) < Int(T_4)$. Thus, e_2 is called a transition edge.

Definition 5. A tree is a *noise tree* if it has fewer vertices than the given limit value for minimum cluster size. As mentioned above, a tree refers to a segment of the given image. An image segmentation process may generate several small, undesired segments, called noise in [2,7].

In Fig. 1(c), T_7 is assumed to have less than the value of the parameter that indicates the limit for the smallest possible cluster size. Thus, T_7 is a noise tree. For this reason, T_7 should not be separated from T_6, and e_4 should not be cut.

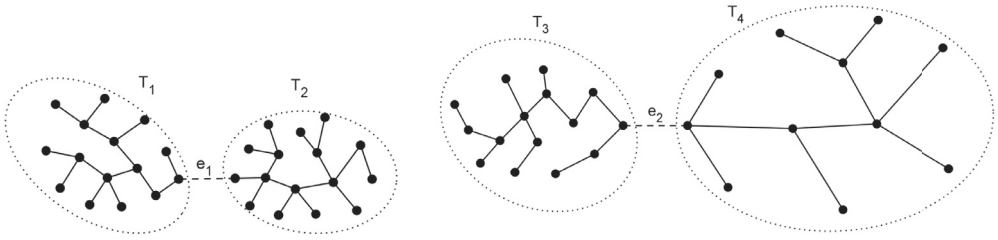
In many clustering and image segmentation experiments, noise segments are a problem. To overcome this problem, some methods, such as [17], use global information and some, such as [15], merge small segments to eliminate noise after the segmentation process is complete. In the proposed method, noise segments are eliminated during the segmentation process.

4. Foundations of our method

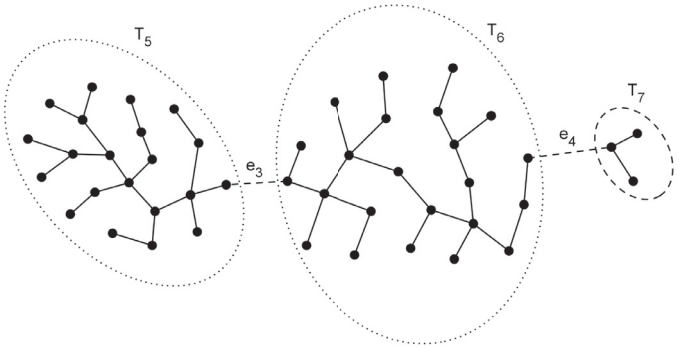
4.1. Prim's sequential representation of MST

Definition 6. For a graph $G = (V, E)$, a Prim's sequential representation of the MST (PSR-MST or simply PSR) can be expressed as the $T_{PSR}(v_s, V_T, Q_E, Q_W)$ of the MST $T = (V, E_T)$ of the graph G, where v_s is the starting vertex (selected arbitrarily), the vertex set is V_T, the edge queue is Q_E, the weight queue is Q_W (which is initially empty, similar to the notation in [23]).

The time complexity of Prim's algorithm is $O(|E| \log |V|)$ [22]. The process of finding the closest vertex of the tree for the number of vertices V consumes a significant amount of time. If the



(a) e_1 is an *explicit edge* (b) e_2 is a *transition edge*



(c) T_7 is a *noise tree*

Fig. 1. Examples of inconsistent edges (e_1, e_2, e_3 and e_4) and noise tree T_7.

Algorithm 1: Steps of forming the PSR-MST.

Input : $G = (V, E)$ – the given graph
Output: $T_{PSR}(v_s, V_T, Q_E, Q_W)$ – the PSR-MST to be extracted from the graph G

1 Initialize $V_T \leftarrow \varnothing$, $Q_E \leftarrow \varnothing$ and $Q_W \leftarrow \varnothing$
2 Specify a starting vertex v_s in V arbitrarily
3 Add v_s to V_T
4 Choose an edge (u, v) that is a minimally weighted edge where $u \in V_T$ and $v \notin V_T$
5 Add v to the vertices set V_T
6 Insert (v, u) at the back of the queue Q_E
7 Insert $w(u, v)$ at the back of the queue Q_W
8 If $|V_T| < |V|$, go to step 4

Fibonacci heap is used in the algorithm of PSR-MST, the computational time is reduced to $O(|E| + |N|\ log\ |V|)$ [30,31]. The Fibonacci algorithm adds the edge sorting order and excludes edges that lead to a loop [32]. Therefore, in this paper, the Fibonacci heap is used with Prim's algorithm to reduce the computational time.

According to Prim's algorithm [22], first, v_s is added to V_T. Then, the smallest edge (u, v), such that $u \in V_T$ and $v \notin V_T$, is added to V_T. At the same time, $e_i^Q = (u, v)$ is inserted at the end of Q_E and $w_i^Q = w(u, v)$ is inserted at the end of Q_W. $i = 1 : n - 1$ defines the index numbers range in queues Q_E and Q_W, where $n = |V|$. e_i^Q refers to the ith element in Q_E; similarly, its weight value w_i^Q refers to the ith element in the Q_W. From this point of view, the index of e_i^Q is equal to the index of w_i^Q. The procedure continues until $|V_T| = |V|$. The algorithm is briefly sketched in Algorithm 1 [23].

4.2. Sequential clustering method and Wang et al.'s cutting criterion

Definition 7. An edge queue Q_E can also be assumed to be an edge string E_T. E_1 and E_2 are two substrings that consist of edges extracted from either side of an index of the queue Q_E in the same order as in Q_E for a given length [23].

The predetermined value of the parameter l specifies the substring length, which is defined to be $l > 0$. We set the length of E_1 and E_2 such that $|E_1| = l$ and $|E_2| = l$.

In this approach, the segmentation of an image is performed by cutting an edge on the MST of the image, as done in [17]. The main problem here is determining whether an edge is an inconsistent edge or not. In our method, to answer this question, the edge e_i is compared to two substrings, i.e., E_1 and E_2, on either side in the queue Q_E. The edge e_i to be examined is subsequently selected from Q_E using the following sequence: $i = l + 1, l + 2, \ldots, |Q_E| - l$ [23]. Several different comparison criteria can be used. In [23], the following criterion is used:

$$w(u, v) > max\ (max(E_1),\ max(E_2)) \tag{1}$$

As seen in Algorithm 2, each edge that satisfies the criterion is inserted at the end of an inconsistent edge queue. Finally, these edge queues are removed from the MST.

Algorithm 2: The steps of Wang et al.'s sequential clustering method [23].

Input : $T_{PSR}(v_s, V_T, Q_E, Q_W)$ and the parameter l
Output: Inconsistent edge queue Q_I

$Q_I \leftarrow \varnothing$
for $i \leftarrow l + 1$ **to** $|Q_E| - l$ **do**
  $w_{max1} \leftarrow max\ (Q_{w,\ i-l,\ i-1})$
  $w_{max2} \leftarrow max\ (Q_{w,\ i+1,\ i+l})$
  if $w_i^Q > max\ (w_{max1},\ w_{max2})$ **then**
    Insert e_i^Q at the end of the queue Q_I
    $i \leftarrow i + l$

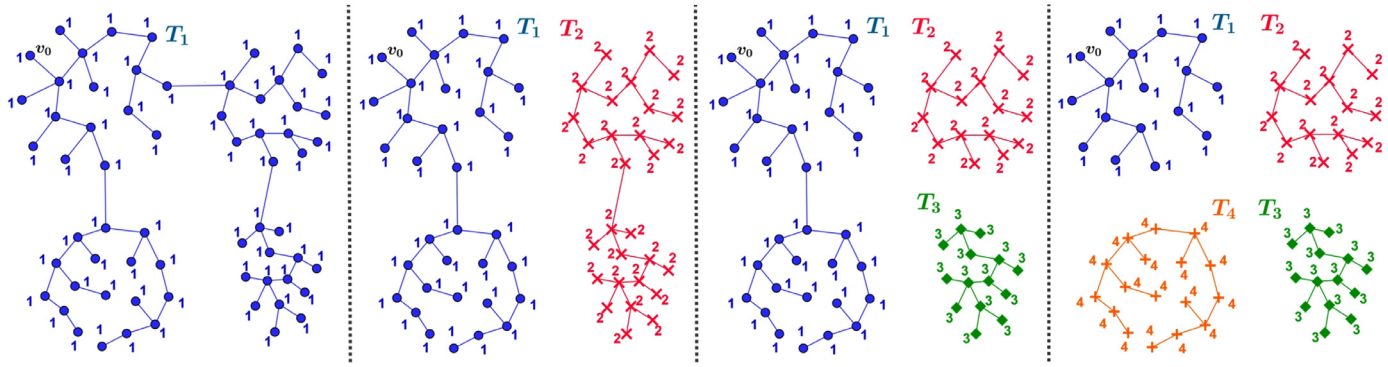


Fig. 2. The steps of separating the tree T into subtrees. Labels 1–4 on the vertices refer to which tree a vertex belongs to.

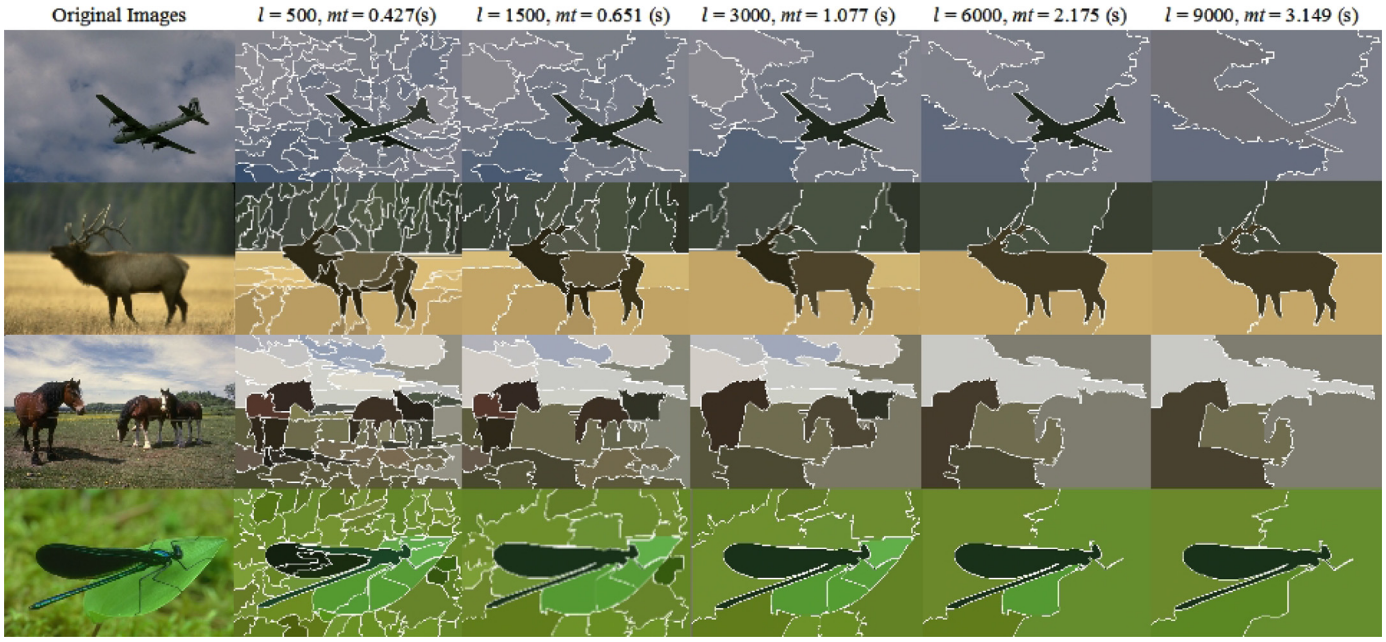


Fig. 3. Segmentation results of some of the images in the BSDS300 using Wang et al.'s sequential clustering method (σ =0.5).

4.3. Felzenszwalb and Huttenlocher's merging criterion

Wang et al.'s criterion determines the explicit edges to be inconsistent edges; however, it ignores transition edges. To solve the problem of identifying transition edges, in [15], a comparison criterion is presented as follows:

$$w(u,v) < min\ (Int(C_1) + \tau(C_1),\ Int(C_2) + \tau(C_2)) \tag{2}$$

In Eq. (2), $Int(C_1)$ and $Int(C_2)$ are the largest weights in the clusters C_1 and C_2, respectively. $\tau(C_1)$ and $\tau(C_2)$ are threshold values that are set by taking into account the size of the clusters C_1 and C_2 such that the formula is defined as follows [15]:

$$\tau(C) = k\ /\ |C| \tag{3}$$

In Eq. (3), k is a constant that controls how much greater the difference between two clusters must be than their internal differences. $|C|$ refers to the size of the cluster C, i.e., the number of elements in C.

5. Our segmentation method

In our method, the substrings E_1 and E_2 in Eq. (1) are taken into account, instead of the clusters in Eq. (2), because the method in [23] (which is the basis of our method) cuts edges on an existing tree, whereas the method in [15] merges clusters to create trees. The criterion in Eq. (1) determines whether an edge should be cut or not, whereas the criterion in Eq. (2) determines whether two clusters should be merged or not. However, the criterion in Eq. (2) considers transition edges and explicit edges, whereas the criterion in Eq. (1) considers only explicit edges. Moreover, the criterion in Eq. (2) uses a threshold value for adaptation. Thus, the two criteria in Eqs. (1) and (2) are utilized in our method as follows:

$$w(u,v) > min\ (max(E_1),\ max(E_2)) + c \tag{4}$$

In Eq. (4), c is a threshold value that is one of the two given parameters; it adjusts how different an inconsistent edge is from nearby edges on any side. Using the formula in Eq. (5), we obtain an appropriate threshold value for each image, i.e., the value c [33]. Using the formula, the gradient of the edge weight queue is evaluated, because the weight of an edge is compared to the weights of the edges around it. In this way, we automate the process of determining the parameter c.

$$c = \frac{\sum_{i=1}^{|Q_W|-1} |w_{i+1}^{Q} - w_i^{Q}|}{|Q_W| - 1} \tag{5}$$

In Eq. (5), Q_W is the queue of the weight values of the edges on the edge queue Q_E. $|Q_W|$ is the size of the queue Q_W. w_i^Q is the value of the ith element in Q_W and w_{i+1}^Q is the value of the next element of w_i^Q.

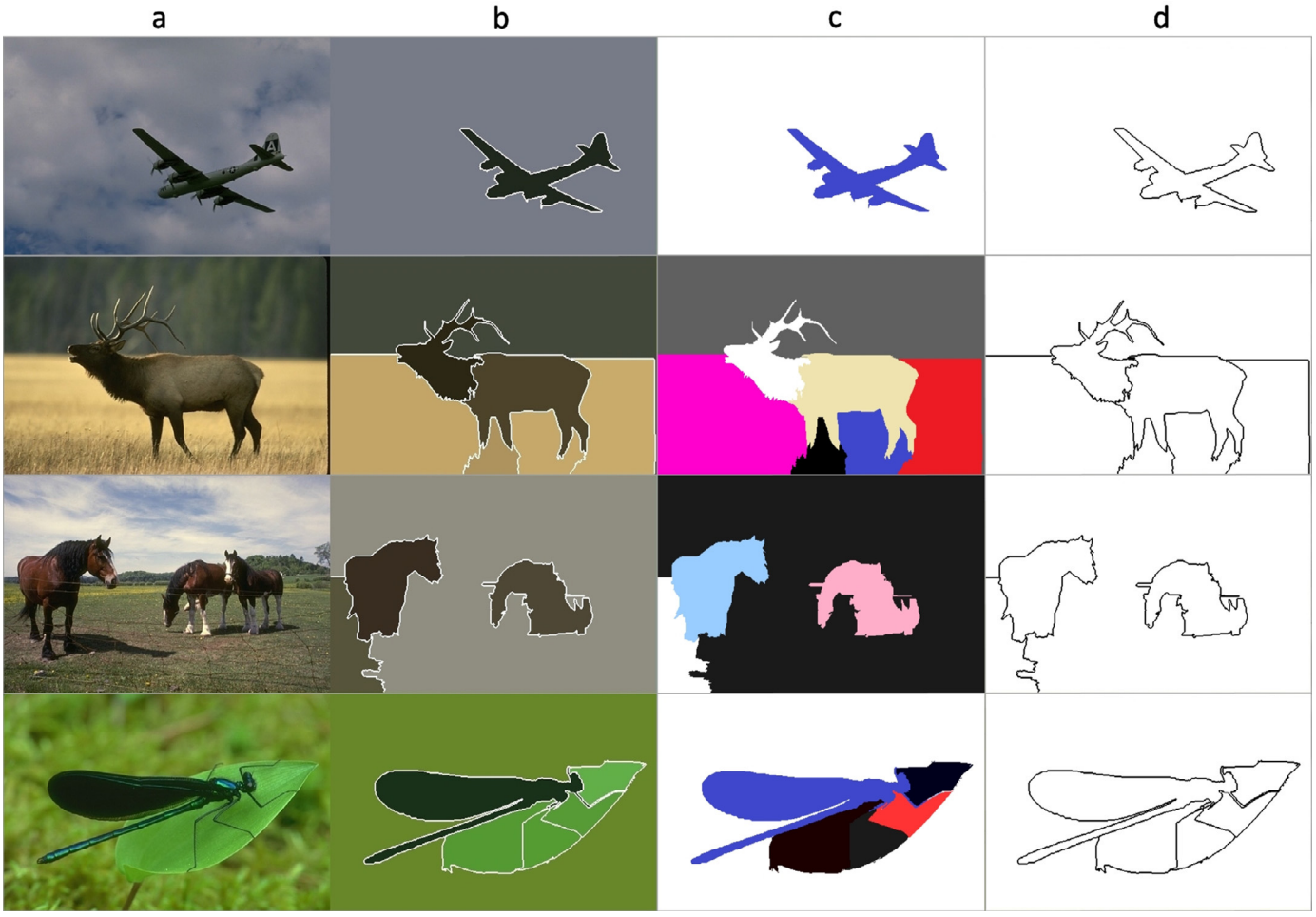


Fig. 4. (a) Some of the original images in the dataset and the presentations of the segmentation results of our method with (b) mean color value, (c) random color value, and (d) boundary lines ($\sigma = 0.5$, $l = 2000$, $mt = 0.68$ s). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

The parameter l in Algorithm 3 specifies both the number of edges to be compared with e_i and the possible minimum size of a segment in the segmentation result.

Algorithm 3: The segmentation steps of our algorithm.

Input : $T_{PSR}(v_s, V_T, Q_E, Q_W)$ and the parameter l
Output: Tree set $T = T_1, T_2, \ldots, T_t$

$sum \leftarrow 0$
for $i \leftarrow 1$ **to** $|Q_W| - 1$ **do**
  diff $\leftarrow |w_{i+1} - w_i|$
  $sum \leftarrow sum +$ diff
$c \leftarrow sum \ / \ (|Q_W| - 1)$
$t \leftarrow 1$
label $T(v_s)$ as T_t
for $i \leftarrow l + 1$ **to** $|Q_E| - l$ **do**
  $w_{max1} \leftarrow max\ (Q_{w,\ i-l,\ i-1})$
  $w_{max2} \leftarrow max\ (Q_{w,\ i+1,\ i+l})$
  $(u, v) \leftarrow e_i^Q$
  if $w_i^Q > min\ (w_{max1},\ w_{max2}) + c$ **then**
    if $|T(u)| > l$ *and* $|T(v)| > l$ **then**
      cut (u, v)
      $t \leftarrow t + 1$
      label $T(v)$ as T_t
      $i \leftarrow i + l$

For any vertex v_i, $T(v_i)$ refers to the tree to which the vertex v_i belongs. As the tree $T(v_i)$ is labeled as T_t, the vertices in the tree $T(v_i)$ are labeled using t. This choice shows that vertices in a tree are labeled using the same value that denotes the tree to which a vertex belongs, and has a link to the same root node. Each root node includes some information about the tree to which it belongs, such as the number of vertices in the tree (the size of the tree). In this paper, T_t indicates a tree; its size is expressed as $|T_t|$. If the size of at least one of the subtrees on the end points of the edge is less than the given length value l, the edge is not cut. If an inconsistent edge is cut, a new tree is created, and its vertices are labelled with new values. The step-by-step labeling process can be seen in Fig. 2.

A broad overview of the flow is shown in Algorithm 3.

6. Experimental results

We tested our method on the BSDS300 of the Berkeley Segmentation Dataset [24]. This dataset has 200 unique images for setting our parameter scale, and 100 images for testing our method. The dimensions of these images are 481 × 321 or 321 × 481 pixels. We coded the program using the C++ programming language and used the techniques of object oriented programming (OOP). To import, illustrate, and export the images and their results, we used the OpenCV (open source computer vision) library [34]. We ran the program on a computer with an Intel(R) Core(TM) i5 1.70-GHz CPU and 6 GB of RAM.

Real images are generally ill posed, which can misdirect the segmentation process and the boundaries may be violated. To

Table 1
Boundary benchmarks of Wang et al.'s method and our method on the Berkeley Segmentation Dataset (BSDS300).

Method	σ	l	Best l	ODS	OIS	Mean time (ODS) (s)
Wang et al.'s [23]	0	100–1500[a]	100	0.49	0.50	0.30
Ours	0	100–2000[a]	200	0.55	0.57	0.37
Wang et al.'s [23]	0.5	200–3100[a]	800	0.52	0.56	0.49
Ours	0.5	100–2000[a]	200	0.57	0.59	0.41
Wang et al.'s [23]	1.2	300–4200[a]	1100	0.54	0.58	0.55
Ours	1.2	100–2000[a]	200	0.59	0.61	0.39

[a] By 100 increment.

reduce the level of this problem, we use a 5×5 Gaussian smoothing filter as a pre-processing method to reduce image noise and redundant details [35]. In Figs. 3 and 4, we used a Gaussian filter with $\sigma = 0.5$. We first directly implemented Wang et al.'s sequential data clustering method [23] for image segmentation, and then, tested it on the dataset. The algorithm has only one parameter; as its value increases, the processing time increases, as can be seen in Fig. 3. Controlling the process using the parameter l is very complicated, because the parameter varies over a large range. In Fig. 3, the results for four images from the dataset are shown. On the top of each column, the value of the parameter l and the mean time (mt) of the segmentation process for four images for the related columns are presented. The average duration of the segmentation process is expressed in seconds (s).

We also apply our method to images that are the same as those in Fig. 3. For these images in Fig. 4, we use the same parameter value of $l = 2000$ and the same Gaussian filter with $\sigma = 0.5$ for the pre-processing stage. We separately and automatically chose the value of the parameter c based on Eq. (5) for each image, as shown in Algorithm 3. The results may be better if different parameters are used for each image. We used the same parameters for every image in Fig. 4.

As seen in Fig. 4, our results are visually better than the base clustering algorithm. The advantages of our method are that it considers transition edges in addition to explicit edges, it applies a threshold that can adapt to images automatically, and it takes noise trees into account. However, this segmentation processes can be misdirected in some parts of the images. Because both of these methods focus on the local features of images, rather than global features, some regions in images might be locally similar to each other. This also causes undesired MST structures to begin to form. This situation causes both methods to depend on the pre-processing technique used. The stages of the segmentation of an image are presented in Fig. 6, which shows that in the stages, certain local similarities will misdirect the segmentation process.

First, we tuned the scale of parameter l using 200 training images and tested the methods for all the 100 test images of the BSDS300. To obtain benchmark results, we used a boundary-benchmarking program from [36]. The program provides a summary score as the best F-measure on the dataset for the optimal dataset scale (ODS) for the entire dataset and the optimal image scale (OIS) for each image. The F-measure is the harmonic mean of precision and recall [36]. We tested our method and Wang et al.'s method for different σ parameter values of a Gaussian pre-processing filter. In Table 1, the results for ODS and OIS, and the scales used, can be seen for the two methods. In this benchmark, a value range for the parameter is needed. In Fig. 5, the precision and recall curves of these methods can be seen.

The mean processing time of our method is slightly longer than Wang et al.'s method. Because, our method also calculates the formula in Eq. (5) for the c threshold value and the sizes of the trees to be separated. However, Wang et al.'s method provides the best of its dataset benchmark scores (ODS) at higher l values than for

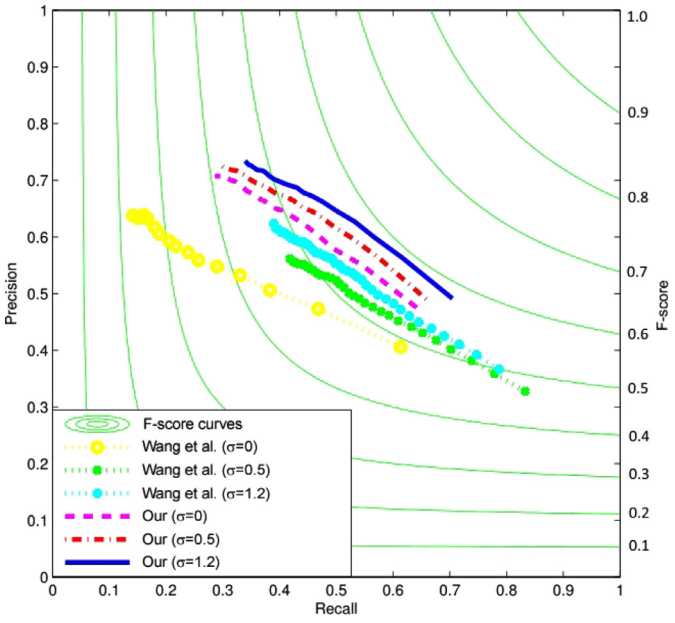


Fig. 5. Precision and recall curves of boundary benchmarks of Wang et al.'s method and our method.

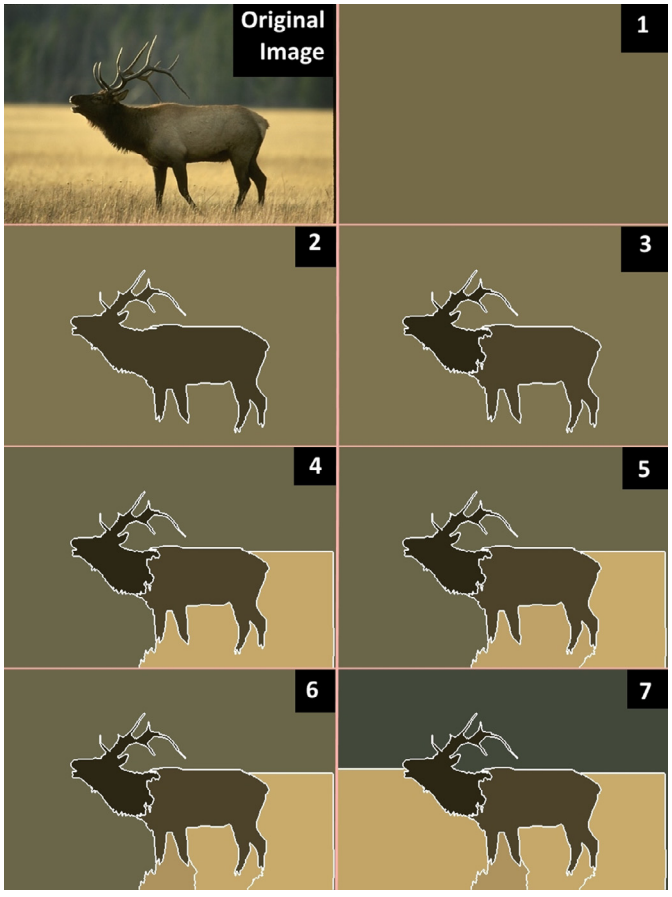


Fig. 6. The original image and versions based on the mean color value of the stages [1–7] of sequential segmentation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Boundary benchmarks of our method and currently popular image segmentation methods for the Berkeley Segmentation Dataset (BSDS300).

Method	ODS	OIS
Mean Shift [37]	0.63	0.66
NCuts [38]	0.62	0.66
Our method	0.59	0.61
Efficient graph-based [15]	0.58	0.62
SWA [39]	0.56	0.59

ours in most cases. As the value of the parameter l increases, the processing load increases, because of the computation of the maximum weight value of the edge strings for the length l. Therefore, the mean time of Wang et al.'s method is higher than ours for $\sigma = 0.5$ and $\sigma = 1.2$ in Table 1.

Finally, we compare our method to the most popular image segmentation methods, arranged in order of performance, in Table 2. These scores have been taken from [36].

Table 2 shows that our method can compete with the most popular image segmentation methods. Our method does not operate on eigenvectors or textures, and does not perform any processes after the segmentation process, such as merging noise segments. In our method, noise segments are merged during image processing. For these reasons, our method performs fast image processing.

7. Conclusion

Image segmentation is a very important stage for obtaining more meaningful information in high-level image processing applications. Image segmentation is also used in real-time image-processing applications. Thus, both accuracy and speed are important in image segmentation methods. The MST structure is commonly used with a graph-based algorithm for ease of use and speed. We discussed some of the MST-based image-segmentation and data-clustering algorithms. We utilized a novel and successful data clustering method based on Prim's MST representation for image segmentation. As the parameter l of the clustering method increases, the number of clusters is reduced and the elapsed time increases. As shown in this paper, it is difficult to control the segmentation as required. Our parameter c supplies an adaptivity to images that is not outweighed by the increase in the process time. This parameter also detects the transition edges on a graph that are beside explicit edges. In addition to automatic operation, the c parameter can be manually entered. As the parameter c increases, the number of segments is reduced, similar to parameter l. We tested both the classical data clustering method and the novel modified form. The developed method was competitive with the popular segmentation algorithms. Our algorithm offers good performance and a low processing time. However, our method focuses on the local features of images, which may cause incorrect segmentation in some regions. To prevent incorrect segmentation, we utilize a smoothing filter during pre-processing. In future, more qualified pre-processing methods can be developed for this segmentation method. Furthermore, better threshold methods can be developed for the parameter c, or for collecting more global information about the given image.

Acknowledgments

This study was supported by Selcuk University OYP Coordination and Scientific Research Project of Selcuk University.

References

[1] N.R. Pal, S.K. Pal, A review on image segmentation techniques, Pattern Recognit. 26 (9) (1993) 1277–1294.
[2] O.J. Morris, M.D.J. Lee, A.G. Constantinides, Graph theory for image analysis : an approach based on the shortest spanning tree, IEE Proc. Commun. Radar Signal Process. 133 (2) (1986) 146–152.
[3] W. Tao, H. Jin, Y. Zhang, Color image segmentation based on mean shift and normalized cuts, IEEE Trans. Syst. Man Cybern. Part B 37 (5) (2007) 1382–1389.
[4] A.N. Skurikhin, Hierarchical image feature extraction by an irregular pyramid of polygonal partitions, in: Proceedings of ASPRS, Baltimore, Maryland, 2009.
[5] H.D. Cheng, X.H. Jiang, Y. Sun, J. Wang, Color image segmentation: advances and prospects, Pattern Recognit. 34 (12) (2001) 2259–2281.
[6] S. Patil, A.A. Junnarkar, Overview of colour image segmentation techniques, Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3 (9) (2013) 418–423.
[7] Y. Xu, E. Uberbacher, 2D image segmentation using minimum spanning trees, Image Vis. Comput. 15 (1997) 47–57.
[8] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999) 264–323.
[9] M. Wertheimer, Laws of organization in perceptual forms, A Source Book of Gestalt Psychology, 1923, Translation published in Ellis, 1938, pp. 71–88.
[10] B. Peng, L. Zhang, D. Zhang, A survey of graph theoretical approaches to image segmentation, Pattern Recognit. 46 (3) (2013) 1020–1038.
[11] E.K. Lloyd, J.A. Bondy, U.S.R. Murty, Graph Theory with Applications, vol. 62, The Mathematical Gazette, 1978.
[12] Y. Zhou, H. Cheng, J.X. Yu, J. Xu Yu, Graph clustering based on structural/attribute similarities, Proc. VLDB Endow. 2 (2009) 718–729.
[13] C. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters, IEEE Trans. Comput. C-20 (1) (1971) 68–86.
[14] G. Karypis, E.-H. Han, V. Kumar, Chameleon: hierarchical clustering using dynamic modeling, Computer 32 (8) (1999) 68–75.
[15] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation, Int. J. Comput. Vis. 59 (2) (2004) 167–181.
[16] P. Fränti, O. Virmajoki, V. Hautamäki, Fast agglomerative clustering using a k-nearest neighbor graph, IEEE Trans. Pattern Anal. Mach. Intell. 28 (11) (2006) 1875–1881.
[17] Y. Xu, V. Olman, D. Xu, Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees., Bioinformatics 18 (4) (2002) 536–545.
[18] Y. Haxhimusa, W. Kropatsch, Segmentation Graph Hierarchies, Lecture Notes in Computer Science, 3138, Springer-Verlag, 2004, pp. 343–351.
[19] T. Harju, Lecture notes on graph theory, Phys. Rev. Lett. 107 (8) (2011) 085504.
[20] O. BorÅvka, O jistém problému minimálním, Práce Morav. pÅírodovědecké společnosti 3 (3) (1926) 37–58.
[21] J.B. Kruskal, On the shortest spanning subtree of a graph and the traveling salesman problem, Proc. Am. Math. Soc. 7 (1) (1956) 48-48.
[22] R.C. Prim, Shortest connection networks and some generalizations, Bell Syst. Tech. J. 36 (6) (1957) 1389–1401.
[23] G.W. Wang, C.X. Zhang, J. Zhuang, Clustering with Prim's sequential representation of minimum spanning tree, Appl. Math. Comput. 247 (2014) 521–534.
[24] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings of the Eighth IEEE International Conference on Computer Vision, ICCV 2001, 2, 2001, pp. 416–423.
[25] J.C. Gower, G.J.S. Ross, Minimum spanning trees and single linkage cluster analysis, J. R. Stat. Soc. Ser. C 18 (1) (1969) 54–64.
[26] Y.X.V. Olman, D. Xu, Solving data clustering problem as a string search problem, in: Proceedings of the Conference on Stat. Data Mining Know. Dis., Chapman & Hall CRC, 2003, pp. 417–434.
[27] O. Grygorash, Y. Zhou, Z. Jorgensen, Minimum spanning tree based clustering algorithms, in: Proceedings of the 2006 18th IEEE International Conference on Tools with Artificial Intelligence, ICTAI'06, 2006, pp. 73–81.
[28] C. Zhong, D. Miao, R. Wang, A graph-theoretical clustering method based on two rounds of minimum spanning trees, Pattern Recognit. 43 (3) (2010) 752–766.
[29] C. Zhong, D. Miao, P. Fränti, Minimum spanning tree based split-and-merge: a hierarchical clustering method, Inf. Sci. 181 (16) (2011) 3397–3410.
[30] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, vol. 7, second ed., MIT Press, 2001.
[31] M. Fredman, R. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J. ACM 34 (1987) 596–615.
[32] C. Zhong, M. Malinen, D. Miao, P. Fränti, A fast minimum spanning tree algorithm based on K-means, Inf. Sci. 295 (2015) 1–17.
[33] A. Saglam, Minimum Yayilan Agac Tabanli Sirali Goruntu Bolutleme (Minimum Spanning Tree-based Sequential Image Segmentation) (Master thesis), Selcuk University, Turkey, 2016.
[34] G. Bradski, The OpenCV Library, Dr. Dobb's Journal of Software Tools (2000) http://sourcefouge.net/projects/opencvlibrary (Accessed 30.04.16).
[35] M.S. Nixon, A.S. Aguado, Gaussian averaging operator, in: Feature Extraction & Image Processing for Computer Vision, Elsevier Ltd., 2012, pp. 86–88.
[36] P. Arbeláez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, Tpami 33 (5) (2011) 898–916.
[37] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 1–37.
[38] T. Cour, F. Bénézit, J. Shi, Spectral segmentation with multiscale graph decomposition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2, 2005, pp. 1124–1131.
[39] E. Sharon, M. Galun, D. Sharon, R. Basri, A. Brandt, Hierarchy and adaptivity in segmenting visual scenes., Nature 442 (7104) (2006) 810–813.