remote sensing

Article

Parallel Regional Segmentation Method of High-Resolution Remote Sensing Image Based on Minimum Spanning Tree

Wenjie Lin and Yu Li *

Institute for Remote Sensing Science and Application, School of Geomatics, Liaoning Technical University, Fuxin 123000, China; 471510037@lntu.edu.cn

* Correspondence: liyu@lntu.edu.cn; Tel.: +86-137-0498-0233

Received: 30 January 2020; Accepted: 24 February 2020; Published: 1 March 2020

Abstract: With finer spatial scale, high-resolution images provide complex, spatial, and massive information on the earth's surface, which brings new challenges to remote sensing segmentation methods. In view of these challenges, finding a more effective segmentation model and parallel processing method is crucial to improve the segmentation accuracy and process efficiency of large-scale high-resolution images. To this end, this study proposed a minimum spanning tree (MST) model integrated into a regional-based parallel segmentation method. First, an image was decomposed into several blocks by regular tessellation. The corresponding homogeneous regions were obtained using the minimum heterogeneity rule (MHR) partitioning technique in a multicore parallel processing mode, and the initial segmentation results were obtained by the parallel block merging method. On this basis, a regionalized fuzzy c-means (FCM) method based on master-slave parallel mode was proposed to achieve fast and optimal segmentation. The proposed segmentation approach was tested on high-resolution images. The results from the qualitative assessment, quantitative evaluation, and parallel analysis verified the feasibility and validity of the proposed method.

Keywords: minimum spanning tree; high-resolution image segmentation; minimum heterogeneity rule; multicore parallel processing; regionalized fuzzy clustering method

1. Introduction

Remote sensing image segmentation is a process of partitioning image domain into several meaningful regions and serves as a bridge between remote sensing image and high-level information processing [1,2]. Remote sensing image is a type of data that records spectral information of the earth's surface from a spaceborne sensor. It is a standardized spatial data that depicts the spatial grid (pixel) distribution of spectral information after the surface has reflected solar electromagnetic waves [3,4]. In recent years, with more satellites being sent to space, coupled with improvements in sensor technology, high-resolution remote sensing data has gradually became ubiquitous [5]. With a finer spatial scale, high-resolution images provide more detailed information through abundant geometric, textural, and spectral features, resulting in new challenges to the traditional remote sensing image segmentation methods [6,7]. These challenges include geometric noise caused by peripheral objects, the phenomena of different objects having the same spectral signature, and similar object types shown as having varying spectral signatures in high-resolution images [8]. Improvements in spatial resolution have been accompanied by the rapid growth in data scale. Large-scale high-resolution remote sensed imagery is characterized by the use of massive data, which results in additional difficulty for traditional serial methods to meet speed requirements in data processing [9,10]. Traditional segmentation algorithms, based on spectral feature clustering, are unable to effectively segment high-resolution images. The abundant spatial and geometric information, which determines the spatial and geometric models, must be taken into account in building the segmentation model.

Having massive data makes data the decomposition-based parallel algorithm a realistic choice to solve the complexity of computing time, and this approach has become one of the most effective ways to expand and optimize existing segmentation methods to address the massive remote sensing image processing requirements [11]. Many researchers have proposed various parallel image segmentation methods for large-scale high-resolution images [12–14]. For example, Xing et al. [15] proposed a parallel remote sensing image segmentation method combined with decomposition/merging mode and k-means algorithms based on geospatial cyberinfrastructure (GCI). Under this mode, the large-scale image is decomposed into several blocks, which are parallel-divided into regions. The merging process restores the block segmentation outcome to the whole image segmentation result and considers the merging of boundary region between blocks, which solves the problem of over-segmentation caused by decomposition. However, the k-means segmentation algorithm only considers spectral information and not the spatial and geometric information, resulting in difficulties in solving the complexity of high-resolution images.

As a branch of mathematics, graph theory uses the graph as the primary study object and has the ability to describe the internal relations of vertex sets [16–19]. The remote sensing image representation model is built by mapping the coordinates and spectral information of the pixels into vertices. The adjacency relationship of the pixels is regarded as the connected edges between the vertices, which makes the construction algorithm of edge set simple and fast. The graph theory is widely used in object-oriented image segmentation methods [20–23]. For example, Wang et al. [24] used the minimum spanning tree (MST) model in graph theory as the expression model of unmanned aerial vehicle (UAV) orthophoto image and combined the β uniform stabilization algorithm to construct the region merging criterion. Their approach was able to realize multiscale segmentation for high-resolution images using scale parameters in the model. However, the segmentation model only considers the spectral and spatial information and does not introduce geometric information.

Recently, the region-based fuzzy c-means (FCM) algorithm [25,26] has been widely used in remote sensing image segmentation due to its noise immunity, good expression ability of irregularity target boundary, and the lower complexity of segmentation decisionmaking. According to the relationship between region partitioning and segmentation processing, region-based FCM algorithm includes dynamic- and static-partitioning types [10]. The dynamic-partitioning method can simultaneously achieve the fitting of the target boundary by geometric partition [27–30] and optimal estimation of segmentation parameters, but has the deficiencies of slow convergence, high computing time complexity as it needs to repeatedly divide the image domain during iterations, and poor expression for its elaborate structure of complex irregular target boundary. The static-partitioning method divides the image domain once according to the partitioning rule at the initial stage of the algorithm operation, and then the image segmentation model can be established based on these regions. It only needs to estimate the segmentation parameters and avoids repeatedly dividing the image domain in each iteration, which greatly reduces the running time. However, the segmentation results are dependent on the initial partitioning result. Thus, establishing a partitioning method which effectively expresses the irregularity of object boundary is the key to such kind of methods.

This study proposed a parallel regionalization method for large-scale high-resolution images, which includes two main parts: The parallel minimum heterogeneity rule (MHR) [31,32] and the parallel regionalization optimal segmentation. For the first part, the image was decomposed into several independent blocks by regular tessellation, and the regions of blocks were divided in parallel by the MHR partitioning method. Then, the regions of the whole image domain were obtained by merging blocks. The partitioning procedure was based on the MST representation of block, while the region growing algorithm accounted for shape proportionality. In the second part, a regional hidden Markov random field (RHMRF) model [33] was constructed on the regions. The FCM [34,35] objective function was then built by combining the Kullback–Leibler (KL) divergence and information entropy regularization term. Each parameter of the FCM objective function was solved by partial derivative. The proposed method extended the hidden Markov random field (HMRF) from modeling the neighborhood relationship of pixels to the neighborhood relationship of regions, which more effectively obtained model parameter estimation and optimal image segmentation.

2. Methods

Given a high-resolution image $\boldsymbol{Z} = \{z_n = Z(u_n, v_n); (u_n, v_n) \in \boldsymbol{F}, n = 1, 2, \ldots, N, u_n \in [1, row], v_n \in [1, col]\}$ defined in the image domain $\boldsymbol{F}$, where (u_n, v_n) is the coordinate of the pixel n, n is the pixel index, N is the number of pixels, row and col is the number of rows and columns of the image, respectively, $N = row \times col$, $Z(u_n, v_n)$ is the spatial sampling function, $z_n = \{z_{nb}; b = 1, 2, \ldots, B\}$ represents the spectral measure vector of pixel n, b is the band index, and B is the number of bands. Suppose that $\boldsymbol{Z}$ consists of K homogeneous regions, such that, $\boldsymbol{F} = \{\boldsymbol{D}_k; k = 1, 2, \ldots, K\}$, where k is the index of homogeneous region, and $\boldsymbol{D}_k$ is homogeneous region K. The set of pixels in the homogeneous regions $\boldsymbol{Z}_k = \{z_n; (u_n, v_n) \in \boldsymbol{D}_k\}$ describes the homogeneity of region k. The segmentation algorithm defines the similarity of homogeneous regions and the optimal segmentation of image domain $\boldsymbol{F} = \{\hat{\boldsymbol{D}}_k; k = 1, \ldots, K\}$ under its constraints.

The image domain is decomposed into T blocks by regular tessellation, such that $\boldsymbol{F} = \{\boldsymbol{F}_t; t = 1, \ldots, T\}$, $\boldsymbol{Z} = \{\boldsymbol{Z}_t; t = 1, \ldots, T\}$, subimage $\boldsymbol{Z}_t = \{z_o = Z(u_o, v_o); (u_o, v_o) \in \boldsymbol{F}_t, o = 1, \ldots, N_t, u_o \in [1, row_t], v_o \in [1, col_t]\}$, t is the index of block, o is the local pixel index of block, N_t is the total number of block t, and row_t and col_t are the numbers of rows and columns in block t, respectively. The number of blocks can be computed as:

$$T = T_r \times T_c \tag{1}$$

where T_r is the rows of blocks, and T_c is the columns of blocks.

$$T_r = \left\lceil \frac{row}{g} \right\rceil, T_c = \left\lceil \frac{col}{g} \right\rceil \tag{2}$$

where $\lceil \bullet \rceil$ stands for ceil operation, g × g is the tessellation parameter.

2.1. Parallel MHR Method

2.1.1. Block Partitioning

For subimage $\boldsymbol{Z}_t$, its image domain $\boldsymbol{F}_t$ can be divided into ${I_t}^S$ disjoint nonempty regions according to the MHR, such that, $\boldsymbol{F} = \{\boldsymbol{F}_{ti}; i = 1, 2, \ldots, {I_t}^S\}$, the corresponding sub-image is $\boldsymbol{Z}_{ti} = \{z_a; a = 1, 2, \ldots, N_{ti}, (u_a, v_a) \in \boldsymbol{F}_{ti}\}$, N_{ti} is the number of pixels in the i-th region of block t, a is the local pixel index of the i-th region of block t, S is the threshold of MHR, and each homogeneous region $\boldsymbol{F}_{ti}$ corresponds to a certain land use and land cover (LULC) class or part of it. In order to better express the explicit and implicit features of blocks, the MST model is used to express blocks as image minimum spanning tree field (IMSTF). Let the pixels of block be mapped to vertices, the 8-neighborhood of pixels is taken as the adjacency relationship, and the nonsimilarity measure between the spectral measure vectors between the pixels is taken as the edge weight, the block can be expressed as graph $\boldsymbol{G'}_t = (\boldsymbol{V}_t, \boldsymbol{E}_t, \boldsymbol{\Psi}_t)$, where $\boldsymbol{V}_t = \{v_a = (u_a, v_a, z_a); a = 1, 2, \ldots, N_t\}$ is the vertex set, $\boldsymbol{E}_t = \{e_c = (v_{c1}v_{c2}); c = 1, 2, \ldots, C_t, v_{c1}, v_{c2} \in \boldsymbol{V}_t\}$ is the edge set, and $\boldsymbol{\Psi}_t = \{\varphi_c = \varphi(v_{c1}, v_{c2}); c = 1, 2, \ldots, C_t\}$ is the edge weights set. In order to improve the computational efficiency, each edge in $\boldsymbol{E}_t$ has to be different. Therefore, the construction method for the edge set was proposed.

As shown in Figure 1, for a vertex $a \in (\{1, 2, \ldots, N_t - row_t\}/\{row_t, 2row_t, \ldots, (col_t-1)\times row_t\})$, which is not in the last row or column, its adjacent edge set is $\boldsymbol{E}_a = \{(v_a v_{a+1}), (v_a v_{a+rowt}), (v_a v_{a+rowt+1}), (v_{a+rowt} v_{a+1})\}$. For a vertex $a \in \{row_t, 2row_t, \ldots, (col_t-1)\times row_t\}$ or $a \in \{(col_t-1)\times row_t+1, (col_t-1)\times row_t+2, \ldots, N_t\}$, which is in the last row or column but not the lower right corner, its adjacent edge set is $\boldsymbol{E}_a = \{(v_a v_{a+rowt})\}$ or $\boldsymbol{E}_a = \{(v_a v_{a+1})\}$, respectively. The adjacent edge set of a vertex in the lower right corner is $\boldsymbol{E}_a = \emptyset$. The edge set $\boldsymbol{E} = \{\boldsymbol{E}_a; a = 1, 2, \ldots, N_t\} = \{\boldsymbol{e}_c = (\boldsymbol{v}_{c1}\boldsymbol{v}_{c2}); c = 1, 2, \ldots, C_t, \boldsymbol{v}_{c1} \in \boldsymbol{V}_t, \boldsymbol{v}_{c2} \in N_p(\boldsymbol{v}_{c1})\}$, where $N_p(\boldsymbol{v}_{c1})$ is the 8-neighborhood of vertex $\boldsymbol{v}_{c1}$, c is the index of edge, C_t is the number of edges, and its calculation equation as follows:

$$\begin{aligned} C_t &= 4(row_t - 1) \times (col_t - 1) + col_t - 1 + row_t - 1 \\ &= 4N_t - 3(row_t + col_t) + 2 \end{aligned} \tag{3}$$

$\varphi_c = \varphi(\boldsymbol{v}_{c1}, \boldsymbol{v}_{c2})$ is defined as the nonsimilarity measure of two vertices spectral vectors connected by $\boldsymbol{e}_c$:

$$\varphi_c = \frac{\exp(t_1 \times d_{c1c2})}{\cos(t_2 \times \theta_{c1c2})} \tag{4}$$

where d_{c1c2} and θ_{c1c2} are the euclidean distance and angle of the spectral vector between vertex $\boldsymbol{v}_{c1}$ and $\boldsymbol{v}_{c2}$, respectively. These parameters can be expressed as:

$$\theta_{c1c2} = \arccos\left(\sqrt{\frac{z_{c1}^T z_{c2}}{\left(z_{c1}^T z_{c1}\right) \times \left(z_{c2}^T z_{c2}\right)}}\right) \tag{5}$$

$$d_{c1c2} = \sqrt{(z_{c1} - z_{c2})^T (z_{c1} - z_{c2})} \tag{6}$$

where t_1, $t_2 \in (0,1)$ are the coefficients of d_{c1c2} and θ_{c1c2} in Equation (4), respectively. After a lot of experiments, we found that the fine results could be obtained by setting $t_1 = 0.1$ and $t_2 = 0.8$.

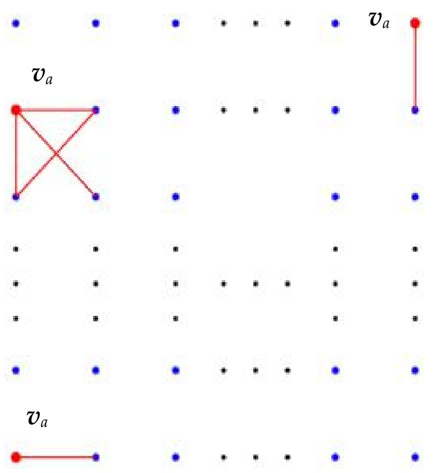


Figure 1. Calculation method of adjacent edge set $\boldsymbol{E}_t$.

For the block graph $\boldsymbol{G}'_t$, the MST rule was introduced to simplify $\boldsymbol{G}'_t$, and $\boldsymbol{T}_t = (\boldsymbol{V}_t^-, \boldsymbol{E}_t^-, \boldsymbol{\Psi}_t^-)$ was obtained. The vertex set $\boldsymbol{V}_t^- = \boldsymbol{V}_t$, edge set $\boldsymbol{E}_t^- = \{\boldsymbol{e}_g = (\boldsymbol{v}_{g1}\boldsymbol{v}_{g2}); g = 1, 2, \ldots, N_t - 1, \boldsymbol{v}_{g1}, \boldsymbol{v}_{g2} \in \boldsymbol{V}_t^-\}$, and $\boldsymbol{\Psi}_t^- = \{\varphi_g = \varphi(\boldsymbol{v}_{g1}, \boldsymbol{v}_{g2}); g = 1, 2, \ldots, N_t - 1\}$ is the set of connection weights corresponding to the connected edges. φ_g is defined by Equation (4). $\boldsymbol{T}_t$ has the following advantages:

(1). The operation time complexity based on $\boldsymbol{T}_t$ is $O(|\boldsymbol{V}_t^-|)$, which has linear complexity.

(2). Every edge of $\boldsymbol{E}_t^-$ is a cut edge, such that deleting or restoring an edge will increase or decrease a region, which greatly simplifies the complexity of the partition operation, and each region corresponds to a tree.

(3). The boundary of the target on $\boldsymbol{T}_t$ is generated along the edge, which is helpful to express the complex and irregular boundary of the target.

The classical algorithms for solving MST are the Prim [36] and Kruskal [37] algorithms, which are both based on the greedy strategy. The Prim algorithm is a vertex-driven mode with time complexity of $O(|\boldsymbol{E}| + |\boldsymbol{V}|\log|\boldsymbol{V}|)$ based on the Fibonacci heap and is suitable for dense graphs. Kruskal algorithm is an edge-driven algorithm with time complexity of $O(|\boldsymbol{E}|\log|\boldsymbol{E}|)$, and is appropriate for sparse graphs. Generally speaking, the graph is called sparse graph when $0 \leq |\boldsymbol{E}| \leq |\boldsymbol{V}|\log|\boldsymbol{V}|$. Otherwise, the graph is dense. From Equation (3), $\boldsymbol{G'}_t$ is constructed by 8-neighborhood, which can be considered as a sparse graph. Thus, the solution for the block MST in this study was realized using the Kruskal algorithm.

At the start of the Kruskal algorithm, all vertices are treated as an MST. The merging tree operation by ascending order of edge weights is then defined until all MSTs are merged into one. Each merging tree operation ensures that the merged graph is a tree which does not contain any cycles. The algorithm's greedy strategy is embodied in the order of merging, such that the edge set is rearranged in ascending order of edge weights before iteration, so the edges with small edge weights are merged early in the iteration procedure. Traditional cycle detection algorithms used by the Kruskal are inefficient because of their search strategy, e.g., disjoint set algorithm and rank-and-path compression algorithm take time complexity $O(\log|\boldsymbol{V}|)$ [38,39]. For improving the performance, a new cycle detection method based on tree marker vectors was proposed, and its pseudo-code o is presented in Algorithm 1.

Algorithm 1. Proposed image MST generation method based on Kruskal algorithm

```
Input: graph $\boldsymbol{G'}_t = (\boldsymbol{V}_t, \boldsymbol{E}_t, \boldsymbol{\Psi}_t)$
Output: MST $\boldsymbol{T}_t = (\boldsymbol{V}_t^-, \boldsymbol{E}_t^-, \boldsymbol{\Psi}_t^-)$
Initialization: $\boldsymbol{V}_t^- = \emptyset$, $\boldsymbol{E}_t^- = \emptyset$, $\boldsymbol{\Psi}_t^- = \emptyset$, $\boldsymbol{t} = \{1, 2, \ldots, |\boldsymbol{V}_t^-|\}$
1.  do $\boldsymbol{E}_t = sortrows(\boldsymbol{E}_t, \boldsymbol{\Psi}_t, \text{“ascending”})$
2.  for each $\boldsymbol{e}_c = (v_{c1}, v_{c2}) \in \boldsymbol{E}_t$
3.    do if $\boldsymbol{t}(v_{c1}) \neq \boldsymbol{t}(v_{c2})$
4.      do $\boldsymbol{E}_t^- = \boldsymbol{E}_t^- \cup \{e_c\}$, $\boldsymbol{\Psi}_t^- = \boldsymbol{\Psi}_t^- \cup \{\varphi_c\}$, $\boldsymbol{V}_t^- = \boldsymbol{V}_t^- \cup \{v_{c1}, v_{c2}\}$, $\boldsymbol{V}_t^- = unique(\boldsymbol{V}_t^-)$
5.      do $\boldsymbol{t}(\boldsymbol{t} = \boldsymbol{t}(v_{c2})) = \boldsymbol{t}(v_{c1})$// all markers of the vertices $\boldsymbol{t}(v_{c2})$ are changed to $\boldsymbol{t}(v_{c1})$
```

The edge set $\boldsymbol{E}$ was rearranged in ascending order of edge weight in step 1 of Algorithm 1. The subsequent cycle judgment process (steps 4, 5) had no order operation in the edge set. Thus, the $\boldsymbol{E}_t^-$ of the MST was strictly in the ascending order, which is an important characteristic of the Kruskal algorithm. The function *unique* $(\boldsymbol{V}_t^-)$ was used for removing duplicate vertices. According step 3 in Algorithm 1, the cycle detection is very simple and takes constant time complexity, which shows the efficiency of proposed method.

The iteration procedure of the proposed image MST generation method based on the Kruskal algorithm is shown in Figure 2. The red and the black points represent the vertices inside and outside of the $\boldsymbol{V}_t^-$, respectively. The red solid lines and the black dotted lines represent the edges inside and outside of the $\boldsymbol{E}_t^-$, respectively. The red numbers were used to show the dynamic change procedure of the tree marker vector. The tree marker vector $\boldsymbol{t}$ was used to mark which tree each vertex belongs to. In the initial stage, each vertex was considered as a tree, as shown in Figure 2a. During the iteration procedure, the merging process for trees was accomplished by changing the tree marker of the vertices. As shown in Figure 2b, both vertices 1 and 2 belong to the tree marked 1 after changing the vertex marker from 2 to 1. The judgment of cycle becomes very simple after the introduction of the tree marker vector $\boldsymbol{t}$, which only requires the determination of whether the tree markers of the two vertices in the merging edge are the same at each iteration. If they are the same, a cycle will be introduced after the merging operation, and the operation is refused. Oherwise, the operation is accepted.

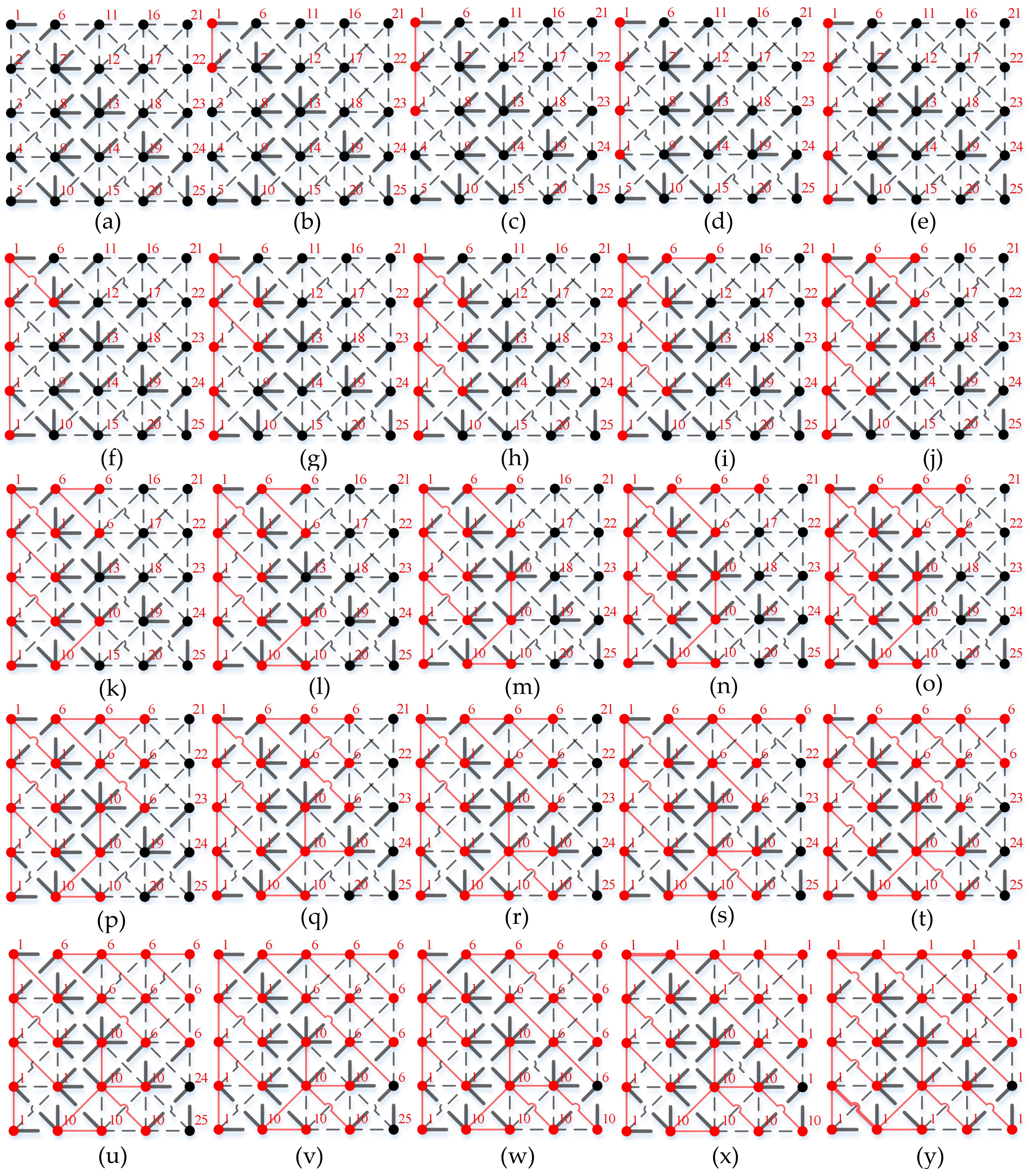


Figure 2. The iteration procedure of the proposed image minimum spanning tree (MST) generation method based on the Kruskal algorithm. (**a**)–(**y**) show the variation of the tree marker vector $\boldsymbol{t}$ with each iteration.

IMSTF reflects the spectral similarity and spatial clustering among pixels and expresses the explicit and implicit features of images. Essentially, it is an extension of the raster structure of image domain and explicitly depicts the connective relationship between vertices, especially its spatial clustering attribute, which has a positive effect on subsequent image processing. On this basis, in order to solve the problems of geometric noise and heterogeneity increase among homogeneous regions in high-resolution images, the similarity of spectral measures and the shape proportionality between regions were taken as the criterion for merging regions based on MHR.

Each vertex was assumed as a homogeneous region. Each edge $\boldsymbol{e}_g \in \boldsymbol{E}_t^-$ was a cut edge and was therefore regarded as the merging edge of its two connected regions. The criterion for merging regions was defined, which used the expression $h(\boldsymbol{F}_{ti}, \boldsymbol{F}_{ti'}) \leq S$ to determine whether the regions merged or not.

In the expression, $\boldsymbol{F}_{ti}$ and $\boldsymbol{F}_{ti'}$ are two regions in the block t connected by $\boldsymbol{e}_g$. Finally, each cut edge was traversed, which was then followed by the merging procedure. The dividing scale S was used to describe the regional heterogeneity.

The merging criterion $h(\boldsymbol{F}_i, \boldsymbol{F}_{i'})$ is a linear combination of spectral similarity and shape proportionality between regions, and is defined as,

$$h(\boldsymbol{F}_{ti}, \boldsymbol{F}_{ti'}) = \alpha h_{\text{color}} + (1-\alpha)h_{\text{shape}} \tag{7}$$

where:

$$h_{\text{color}} = \sum_b \frac{1}{\sigma_b^2}\left[N_{\text{merge}}\sigma_{b,\text{merge}} - \left(N_{ti}\sigma_{b,ti} + N_{ti'}\sigma_{b,ti'}\right)\right] \tag{8}$$

$$h_{\text{shape}} = \gamma h_{\text{compt}} + (1-\gamma)h_{\text{smooth}} \tag{9}$$

$$h_{\text{compt}} = l_{\text{merge}}\sqrt{N_{\text{merge}}} - \left(l_{ti}\sqrt{N_{ti}} + l_{ti'}\sqrt{N_{ti'}}\right) \tag{10}$$

$$h_{\text{smooth}} = N_{\text{merge}}\frac{l_{\text{merge}}}{b_{\text{merge}}} - \left(N_{ti}\frac{l_{ti}}{b_{ti}} + N_{ti'}\frac{l_{ti'}}{b_{ti'}}\right) \tag{11}$$

where $\alpha \in [0, 1]$ is the coefficient of spectral similarity, and h_{color} is the spectral similarity. $\sigma_{b,\text{merge}}$, $\sigma_{b,ti}$, and $\sigma_{b,ti'}$ are the standard deviations of the spectral measure of the merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$), region ti, and ti', respectively. N_{merge}, N_{ti}, and $N_{ti'}$ are the pixels numbers of the merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$), the region ti, and ti', respectively. $1/\sigma_b{}^2$ is the weight of band b and $\sigma_b{}^2$ is the variance of band b. The shape proportionality h_{shape} is a linear combination of the regional shape compactness h_{compt} and the smoothness h_{smooth}, and $\gamma \in [0, 1]$ is the weight of the regional shape compactness. l_{merge}, l_{ti}, and $l_{ti'}$ are the outline lengths of the merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$), region ti, and ti', respectively. b_{merge}, $b_{ti,}$ and $b_{ti'}$ are the regional outer rectangular perimeters of the merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$), region ti, and ti', respectively.

The solving procedure for the regional outer rectangular perimeter is shown in Figure 3a, where a green square represents a pixel in the region, and the regional outer rectangular perimeter is the number of red border squares. Figure 3b,c show the solving procedure of the regional outline. First, an augmented matrix was generated according to the coordinate range of the pixels of the region. As shown in Figure 3b, a green square represents an augmented pixel, and the pixel in the region is marked as 1 (see yellow squares). The element values of the four adjacent pixels were then summed up for each yellow square and used as replacement values. In Figure 3c, the yellow squares, with a value of 4, represent the interior pixels of the region, while the blue squares, with a value less than 4, represent the pixels on the regional outline. The length of the regional outline was determined by the number of blue squares.

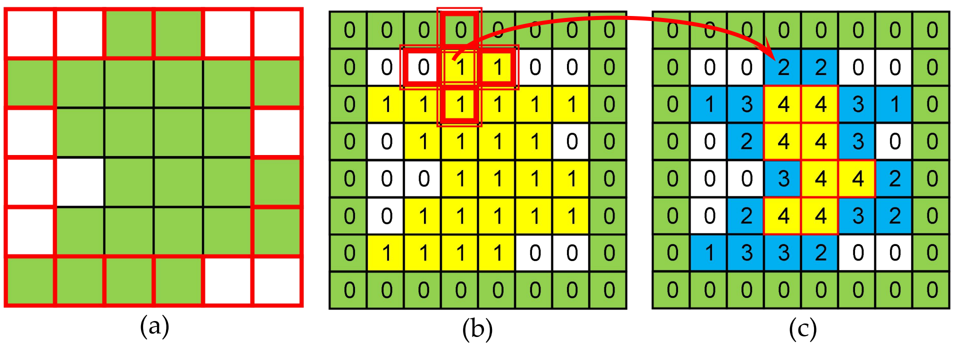


Figure 3. Procedure for the regional outer rectangular perimeter and regional outline. (**a**) shows the outer rectangular perimeter of a region, and (**b**,**c**) show the procedure of region outline.

The merging procedure is presented in Figure 4. Figure 4a,b illustrate the regional outer rectangular perimeter of two regions (b_{ti} and $b_{ti'}$) and the regional outline (l_{ti} and$l_{ti'}$) of region $\boldsymbol{F}_{ti}$ and $\boldsymbol{F}_{ti'}$, respectively. Figure 4c shows the regional outer rectangular perimeter b_{merge} and regional outline l_{merge} of the merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$). The shape complementarity between regions $\boldsymbol{F}_{ti}$ and $\boldsymbol{F}_{ti'}$ was good, so the shape proportionality h_{shape} of merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$) was low, which motivates the merging procedure of two regions ($\boldsymbol{F}_{ti}$ and $\boldsymbol{F}_{ti'}$).

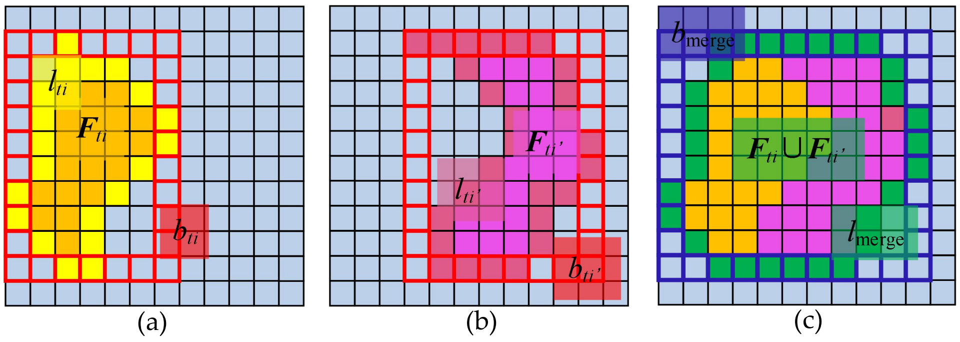


Figure 4. Procedure for merging two regions. (**a**) shows the regional outer rectangular perimeter b_{ti} and regional outline l_{ti} of region $\boldsymbol{F}_{ti}$; (**b**) shows the regional outer rectangular perimeter $b_{ti'}$ and regional outline $l_{ti'}$ of region $\boldsymbol{F}_{ti'}$; (**c**) shows shows the regional outer rectangular perimeter b_{merge} and regional outline l_{merge} of merged region ($\boldsymbol{F}_{ti}\cup\boldsymbol{F}_{ti'}$).

The procedure for the MHR partitioning procedure based on IMSTF is shown in Figure 5. Figure 5a is a simulated-color image, which consists of three homogeneous regions and is marked by three colors: Red, green, and blue. The blue region contains a yellow pixel, which was used to simulate geometric noise. Figure 5b is the MST of the simulated image, which was able to distinctly delineate boundaries. Figure 5c represents the forest obtained by the MHR criterion, with each tree corresponding to a homogeneous region. The MHR describes the regional structure of the image. Based on the simulation results, the MHR based on IMSTF is able to identify the irregular boundaries of the target and deal with the geometric noise effectively.

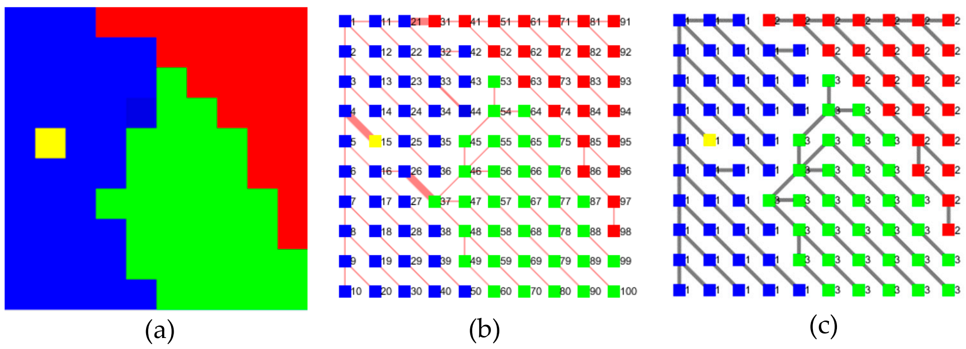


Figure 5. Minimum heterogeneity rule (MHR) partitioning method. (**a**) is a simulated image, (**b**) is the MST of the simulated image, and (**c**) is the result of the MHR partitioning method of the simulated image.

In addition, because $\boldsymbol{E}_t^-$ is arranged in ascending order of edge weights and appears as a disordered hash in space, the smaller edge weight means two regions connected by the edge are more similar in spectral, and the more preferred these two regions will be merged. The procedure of the MHR partitioning method based on the IMSTF can be considered as a procedure for global and progressive region growth. Thus, the problem of the number and spatial distribution of initial growing seed points can be avoided, which is difficult to evade in traditional region growth algorithms. The procedure for the global progressive region growth is similar to the process of simulated annealing. In the initial stage, the regions with the most similar spectral characteristics are preferentially merged to form several pure region growth cores. After more iterations, the complex merging situation (with larger boundary weights) can be judged gradually, and the shape constraint plays a crucial role in making the region grow more compactly.

In terms of block tessellation, assuming each block is independent, each block can be processed in parallel. The parallel task pool $\boldsymbol{Y} = \{y_t = y(\boldsymbol{Z}_t, \boldsymbol{F}_t); t = 1, 2, \ldots, T\}$ is formed by T blocks, where $y(\boldsymbol{Z}_t, \boldsymbol{F}_t)$ is the task of the MHR partitioning method which includes block graph representation, block MST solution, and MHR partitioning, y_t is the result of block partitioning, and t is the task index.

The parallel computing system was made up of P parallel computing units and used to process $\boldsymbol{Y}$ parallel. Since the amount of data in each block was approximately the same, the runtime for each task can also be considered as roughly similar. Therefore, T tasks can be aggregated into P task groups in a balanced task distribution manner. Each parallel computing unit executes its allocated tasks in sequence. Assume that an image is tessellated into 256 blocks with five parallel execution units, which are represented by five color blocks, as shown in Figure 6. The tasks distributed by each parallel computing unit are shown in the grid of color blocks at the left side of Figure 6. Each color block corresponds to a thread in the execution procedure. Threads with the same color formed thread blocks, which were executed in sequence by the corresponding color computing units. After all the blocks were divided into regions, the global result was reduced according to the block index.

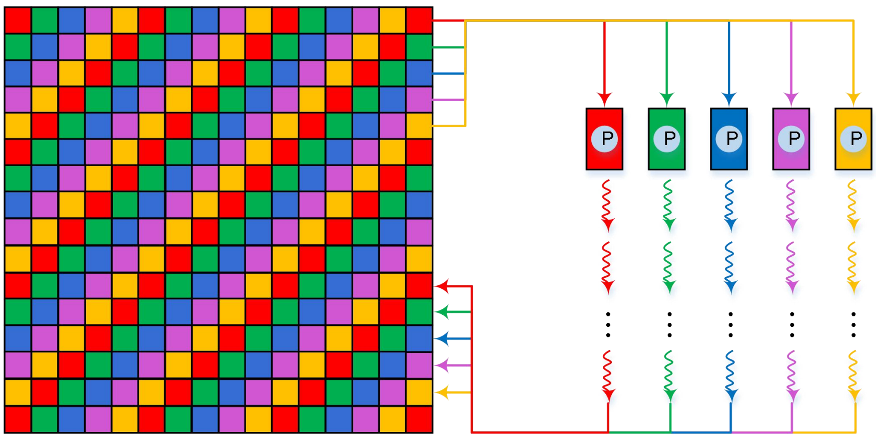


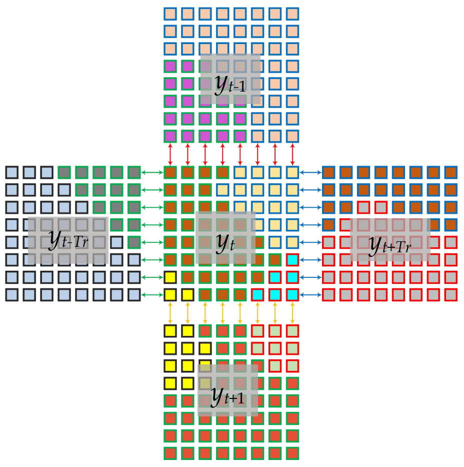
Figure 6. Parallel scheduling of the block MHR partitioning technique.

2.1.2. Block Merging

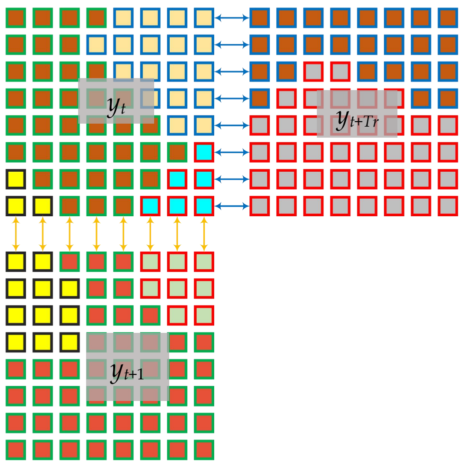
In order to decrease the damage in landscape caused by regular tessellation, the results from block paritioning had to be merged. For the resulting partition y_t of block t, its four neighborhood blocks were y_{t-1}, y_{t+1}, y_{t+Tr}, and y_{t-Tr} (see Figure 7a), where $t = 1, 2, \ldots, T$, and T_r was the row number of the block. The purpose of block merging is to stitch the bordered regions by reconstructing the correlation between adjacent blocks. In merging the block y_t, the neighborhood blocks y_{t-1}, y_{t+1}, y_{t+Tr}, and y_{t-Tr}, were merged pair-wisely to avoid read-write conflicts, resulting in pairs (y_t, y_{t-1}), (y_t, y_{t+Tr}), (y_t, y_{t+1}), and (y_t, y_{t-Tr}).

Figure 7a shows the association between blocks, where the red, blue, yellow, and green bidirectional arrows indicate the associations between y_t and y_{t-1}, y_{t+Tr}, y_{t+1},y_{t-Tr}, respectively. All bidirectional arrows identified by each color form the interblock merge set, which are represented by $\boldsymbol{\Lambda}_{t\_t-1}$, $\boldsymbol{\Lambda}_{t\_t+Tr}$, $\boldsymbol{\Lambda}_{t\_t+1}$ and $\boldsymbol{\Lambda}_{t\_t-Tr}$, respectively. Each bidirectional arrow became the merge edge between two adjacent blocks, such that the red and yellow arrows were vertically correlated, while the blue and green arrows were horizontally correlated. In addition, the squares with different color bordered in Figure 7a represent different homogeneous regions, and the squares with different color fillings represent the divided regions of each block. These homogeneous regions were distributed in different blocks due to the influence of image tessellation. In effect, there was a redundancy problem in the merging procedure.

For example, for the block y_t, its left block y_{t-Tr} and upper block y_{t-1} were merged, with the center on block $y_{t-Tr\text{-}1}$. To avoid this, the actual merging procedure only needed to consider the merging of the central block with its right and lower blocks, as shown in Figure 7b. For the last row or the last column of blocks, only the right and lower blocks had to be merged with the central block (i.e., the lower right block did not need to be merged).



(a)



(b)

Figure 7. Block neighbor relationship. (**a**) represents the interblock merge set of the block and its neighborhood blocks, and (**b**) represents the block basic association structure.

Figure 8 shows the dependence relationship of horizontal interblock merging. For the three blocks y_{t-Tr}, y_t and y_{t+Tr}, as shown in Figure 8a, the merged result of (y_{t-Tr}, y_t) was considered as input for the merging procedure of (y_t, y_{t+Tr}), which means that the merging result of (y_t, y_{t+Tr}) shown in Figure 8c depends on the merged result of (y_{t-Tr}, y_t) shown in Figure 8b. This dependence relationship created difficulties for the parallel merging procedure, which made it necessary to design a better parallel scheduling scheme to decouple the dependence relationship. The dependence relationship of the vertical merging process was similar to the horizontal procedure and is not discussed here.

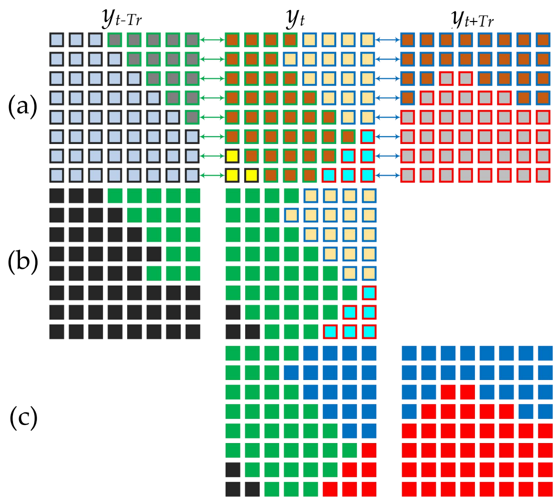


Figure 8. Block merging. (**a**) shows three blocks (y_{t-Tr}, y_t and y_{t+Tr}) without merging procedure; (**b**) shows the merged blocks (y_{t-Tr} and y_t) status; (**c**) shows the merged blocks (y_t, and y_{t+Tr}) status in which block y_t is the merged result in (**b**).

The merging procedure of blocks can be summarized as follows: For each merged edge in $\mathbf{\Lambda}_{t\_t+1}$, if two pixels connected by the edge belong to the same region, continue on with the next merged edge.

Otherwise, determine whether or not to merge these two regions based on the equation $h(\boldsymbol{F}_{ti}, \boldsymbol{F}_{t+1\ i}) \leq S$. If the criterion is satisfied, the two regions are merged. Otherwise, move on to the next merging edge. The flowchart of the algorithm is shown in Figure 9. Using the same principle, the merging set $\boldsymbol{\Lambda}_{t\_t+Tr}$ was assessed, then the process was continued until all blocks were merged.

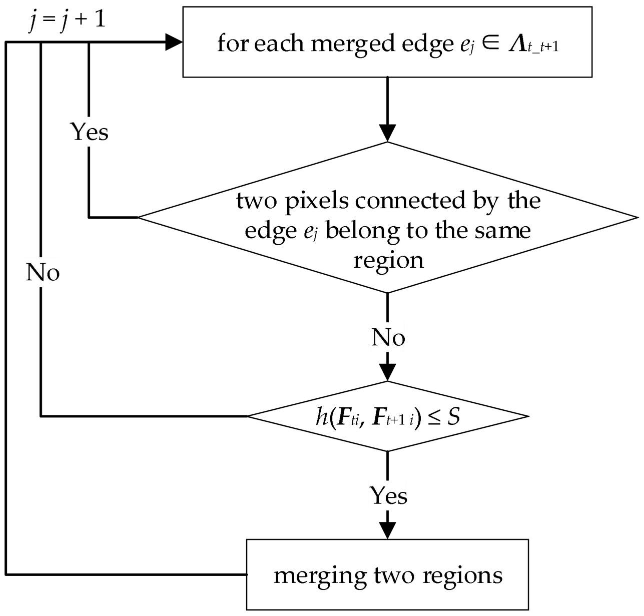

$j = j + 1$
for each merged edge $e_j \in \Lambda_{t\_t+1}$
Yes
two pixels connected by the edge e_j belong to the same region
No
No
$h(\boldsymbol{F}_{ti}, \boldsymbol{F}_{t+1\ i}) \leq S$
Yes
merging two regions

Figure 9. Flowchart of the blocks merging procedure.

Since the merging procedure was carried out serially, when the number of blocks is large, the required processing time will be enormous. To further improve the speed of merging, a hierarchical parallel scheduling method was proposed. The computational dependency diagram for merging blocks is shown in Figure 10a. The computational dependency diagram can be divided into two parts according to the horizontal and vertical merging modes, as shown in Figure 10b,c, respectively. The red bidirectional arrow in Figure 10b indicates a horizontal first layer merging procedure, and the blue bidirectional arrow denotes a horizontal second layer merging procedure. Similarly, the green bidirectional arrow indicates a vertical first layer merging procedure, and the purple bidirectional arrow denotes a vertical second layer merging procedure.

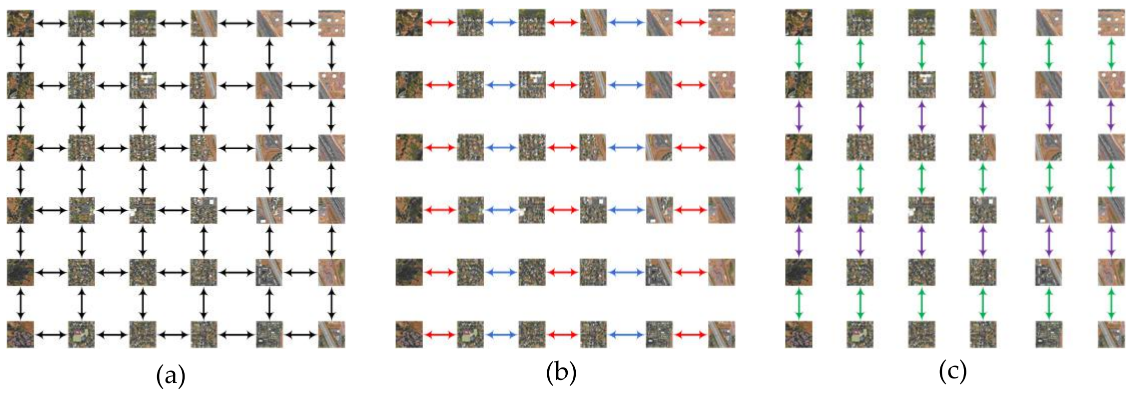


Figure 10. Decomposing of blocks merging dependency. (**a**) is the computational dependency diagram of merging operations between blocks, (**b**,**c**) is the decomposing of blocks merging dependency results by horizontally and vertically, respectively.

The horizontal and vertical merging tasks of Figure 10b,c can be aggregated into the form presented in Figure 11, where Figure 11a,b are the first and second horizontal merging task sets and Figure 11c,d are the first and second vertically merging task sets, respectively. The index sets $\boldsymbol{MI}_1$, $\boldsymbol{MI}_2$, $\boldsymbol{MI}_3$, and $\boldsymbol{MI}_4$ are the block pairs of the first horizontal layer, second horizontal layer, first vertical layer, and second vertical layer, respectively. These index sets can be obtained by the following equations:

$$\boldsymbol{MI}_1 = \begin{cases} \bigcup\limits_{r=1}^{T_r} \bigcup\limits_{c=1}^{\frac{T_c}{2}} \{(2(c-1)T_r + r, (2c-1)T_r + r)\}, \text{ if } \mathrm{mod}(T_c, 2) = 0 \\ \bigcup\limits_{r=1}^{T_r} \bigcup\limits_{c=1}^{\frac{T_c-1}{2}} \{(2(c-1)T_r + r, (2c-1)T_r + r)\}, \text{ else} \end{cases} \tag{12}$$

$$\boldsymbol{MI}_2 = \begin{cases} \bigcup\limits_{r=1}^{T_r} \bigcup\limits_{c=1}^{\frac{T_c}{2}-1} \{((2c-1)T_r + r, 2cT_r + r)\}, \text{ if } \mathrm{mod}(T_c, 2) = 0 \\ \bigcup\limits_{r=1}^{T_r} \bigcup\limits_{c=1}^{\frac{T_c-1}{2}} \{((2c-1)T_r + r, 2cT_r + r)\}, \text{ else} \end{cases} \tag{13}$$

$$\boldsymbol{MI}_3 = \begin{cases} \bigcup\limits_{c=1}^{T_c} \bigcup\limits_{r=1}^{\frac{T_r}{2}} \{((c-1)T_r + 2r - 1, (c-1)T_r + 2r)\}, \text{ if } \mathrm{mod}(T_r, 2) = 0 \\ \bigcup\limits_{c=1}^{T_c} \bigcup\limits_{r=1}^{\frac{T_r-1}{2}} \{((c-1)T_r + 2r - 1, (c-1)T_r + 2r)\}, \text{ else} \end{cases} \tag{14}$$

$$\boldsymbol{MI}_4 = \begin{cases} \bigcup\limits_{c=1}^{T_c} \bigcup\limits_{r=1}^{\frac{T_r}{2}-1} \{((c-1)T_r + 2r, (c-1)T_r + 2r + 1)\}, \text{ if } \mathrm{mod}(T_r, 2) = 0 \\ \bigcup\limits_{c=1}^{T_c} \bigcup\limits_{r=1}^{\frac{T_r-1}{2}} \{((c-1)T_r + 2r, (c-1)T_r + 2r + 1)\}, \text{ else} \end{cases} \tag{15}$$

where T_r and T_c are the row and column numbers of the block, respectively, u = mod(a, b) is the remainder function, a is the dividend, b is the divisor, and u is the remainder. a, b, and u are all integers.

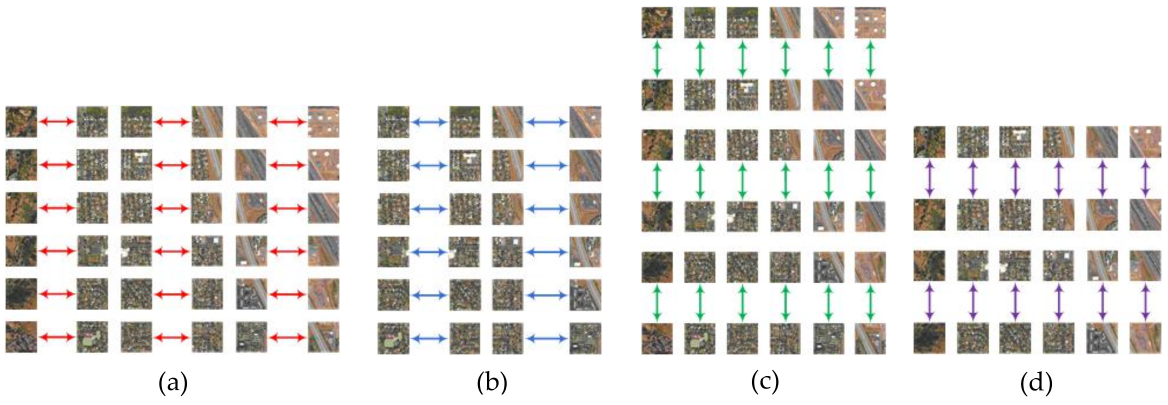


Figure 11. The aggregated result of horizontal and vertical merging tasks. (**a**–**d**) are the first and second horizontal merging task sets, the first and second vertical merging task sets, respectively.

In terms of the aggregated result of the horizontal and vertical merging tasks, the optimized procedure can be realized using a multithread parallel execution mode. Figure 12 presents a horizontal first-level merging parallel scheduling schematic diagram. Five colors represent the five parallel execution units, each of which corresponds to a thread in the execution procedure. Thread blocks with the same color were executed serially by the corresponding color computing units. Horizontal second layer merging, vertical first layer merging, and longitudinal second layer merging were carried out in similar fashion. After all the block merging tasks were completed, the final global results were obtained by reducing according to the corresponding block index.

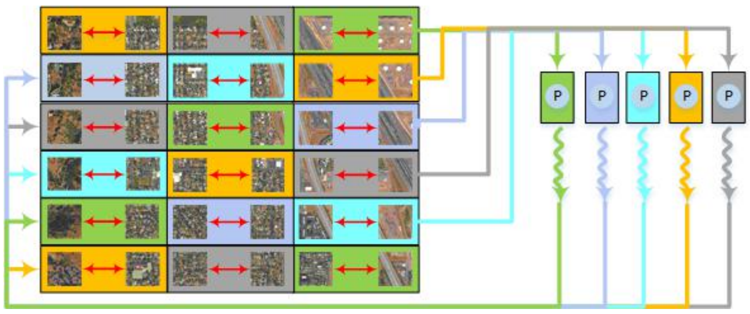


Figure 12. Horizontal first-level merging parallel scheduling schematic diagram.

2.2. *Regional Hidden Markov Random Field–Fuzzy C-Means (RHMRF-FCM) Segmentation*

The parallel MHR partitioning procedure divides the image domain into regions. However, these regions are optimal locally and not necessarily optimal globally. Thus, the general segmentation algorithms need to adopt an additional clustering process to obtain global optimal segmentation results. In this section, the RHMRF model was developed based on the idea of pixel HMRF-FCM image segmentation and using the region as the basic decisionmaking unit.

2.2.1. RHMRF-FCM Segmentation

The image domain $\boldsymbol{F}$ was divided into I^S homogeneous regions using a parallel MHR partitioning method, such that $\boldsymbol{F} = \{\boldsymbol{F}_i; i = 1, 2, \ldots, I^S\}$ is the set of pixels corresponding to each region $\boldsymbol{Z}_i = \{\boldsymbol{z}_a; a = 1, 2, \ldots, N_i\}$, and its label set is $\boldsymbol{L} = \{L_i; i = 1, 2, \ldots, I^S\}$. Then, the objective function of regionalized HMRF-FCM [40] can be written as:

$$F_\lambda\left(\boldsymbol{R}, \theta, \boldsymbol{\eta} \middle| I^S, K\right) = \sum_{i=1}^{I^S}\sum_{k=1}^{K} r_{ik} D_{ik} + \lambda \sum_{i=1}^{I^S}\sum_{k=1}^{K} N_i r_{ik} \log\left(\frac{r_{ik}}{\eta_{ik}}\right) \tag{16}$$

where the first term on the right side (Equation (16)) is the linear weighted sum of information and its fuzzy relations with all the regions, which indicates the cost of label configuration and is the main factor for clustering decisionmaking. The second term refers to the regularization of KL divergence information. It introduces the relationship between the fuzzy relation and the local label of the region, which is the expression of the region's local constraints and can restrain the over-segmentation. $\lambda \in [0,1]$ is the fuzzy degree, which is used to coordinate the relationship between two decision items. $\boldsymbol{R} = \{r_{ik}; i = 1, 2, \ldots, I^S, k = 1,2, \ldots, K\}$ is a fuzzy membership set, where r_{ik} describes the degree which region i belongs to the j-th cluster, and $\boldsymbol{\eta} = \{\eta_{ik}; i = 1, 2, \ldots, I^S, k = 1,2, \ldots, K\}$ is a prior distribution of HMRF model. η_{ik} is the pointwise prior probability of the RHMRF model states [41], which can be expressed as:

$$\eta_{ik} = p(L_i = k | L_{i'}, \boldsymbol{F}_{i'} \in NF_i, \beta) = \frac{\exp\left(-\beta \sum\limits_{\boldsymbol{F}_{i'} \in NF_i} t(L_i, L_{i'})\right)}{\sum\limits_{k'=1}^{K} \exp\left(-\beta \sum\limits_{\boldsymbol{F}_{i'} \in NF_i} t(L_i = k', L_{i'})\right)} \tag{17}$$

where NF_i is the neighborhood relationship of the region $\boldsymbol{F}_i$, i' is the neighborhood region index of $\boldsymbol{F}_i$, k is the labeled index, and $\beta \in [0, 1]$ is a parameter of the clique potentials known as the inverse supercritical temperature. $t(L_i, L_{i'})$ is the indicator function, where $t(L_i, L_{i'}) = 1$ if and only if $L_i = L_{i'}$;

otherwise, $t(L_i, L_{i'}) = 0$. D_{ik} is the information about the class k in the data of region i and can be defined as:

$$\begin{aligned} D_{ik} &= -\log N(\mathbf{Z}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = -\sum_{a=1}^{N_i} \log N(z_a; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= -\sum_{a=1}^{N_i} \log\left\{(2\pi)^{-\frac{B}{2}} |\boldsymbol{\Sigma}_k|^{\frac{1}{2}} \exp\left[-\frac{(z_a-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k)}{2}\right]\right\} \\ \Rightarrow D_{ik} &= -\frac{N_i}{2}[B\log 2\pi + \log|\boldsymbol{\Sigma}_k|] - \frac{1}{2}\sum_{a=1}^{N_i} (z_a-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k) \end{aligned} \quad (18)$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \sum_k\}; k = 1, 2, \ldots, K\}$ is the set of model parameters, $\boldsymbol{\mu}_k$ is the mean vector with B dimension, and $\sum_k$ is the covariance matrix with $B\times B$ dimension. The global optimal segmentation result based on region partitioning is the minimum solution of the objective function (Equation (16)). The explicit class attributes of each region can be obtained by de-fuzzifying the fuzzy relation matrix $\boldsymbol{R}$. At present, defuzzification is usually solved by maximizing the fuzzy relation, such that:

$$L_i = \underset{k}{\operatorname{argmax}}\{r_{ik}; k = 1, 2, \ldots, K\} \quad (19)$$

By substituting Equations (17) and (18) into Equation (16), the final form of the objective function was obtained:

$$\begin{aligned} &F_\lambda\left(\boldsymbol{R}, \theta, \eta \middle| I^S, K\right) \\ &= -\sum_{i=1}^{I^S}\sum_{k=1}^{K} r_{ik}\left\{\frac{N_i}{2}[B\log 2\pi + \log|\boldsymbol{\Sigma}_k|] + \frac{1}{2}\sum_{a=1}^{N_i}(z_a-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k)\right\} + \lambda\sum_{i=1}^{I^S}\sum_{k=1}^{K} N_i r_{ik}\log\left(\frac{r_{ik}}{\eta_{ik}}\right) \end{aligned} \quad (20)$$

where $\boldsymbol{\mu}_k$, $\sum_k$, and r_{ik} are unknown parameters. $\boldsymbol{\mu}_k$ and $\sum_k$ are explicit parameters, such that the partial derivatives of the objective function for these two parameters can be solved by the following equations:

$$\frac{\partial F_\lambda}{\partial \boldsymbol{\mu}_k} = 0 \Rightarrow \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{I^S} r_{ik}\sum_{a=1}^{N_i} z_a}{\sum_{i=1}^{I^S} N_i r_{ik}} \quad (21)$$

$$\begin{aligned} &\frac{\partial F_\lambda}{\partial \boldsymbol{\Sigma}_k} = 0 \Rightarrow \sum_{i=1}^{I^S} r_{ik}\frac{N_i}{2}\left(-\boldsymbol{\Sigma}_k^{-1}\right) + \sum_{i=1}^{I^S} r_{ik}\sum_{a=1}^{N_i}\frac{1}{2}\boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k)^T(z_a-\boldsymbol{\mu}_k)\boldsymbol{\Sigma}_k^{-1} = 0 \\ &\Rightarrow \boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{I^S} r_{ik}\sum_{a=1}^{N_i}(z_a-\boldsymbol{\mu}_k)^T(z_a-\boldsymbol{\mu}_k)}{\sum_{i=1}^{I^S} N_i r_{ik}} \end{aligned} \quad (22)$$

where $\partial\log|\boldsymbol{\Sigma}_k|/\partial\,\boldsymbol{\Sigma}_k = -\boldsymbol{\Sigma}_k^{-1}$, $\partial(z_a-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k)/\partial\,\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}_k^{-1}(z_a-\boldsymbol{\mu}_k)^T(z_a-\boldsymbol{\mu}_k)\boldsymbol{\Sigma}_k^{-1}$.

The fuzzy membership r_{ik} satisfies the constraint condition $\sum_k r_{ik} = 1$, such that the sum of the fuzzy relations of a region belonging to all classes is equal to 1. Therefore, a Lagrange function was constructed to solve the problem based on the constraints,

$$f(r_{i1}, r_{i2}, \ldots, r_{iK}) = \sum_{k=1}^{K} r_{ik}D_{ik} + \lambda\sum_{k=1}^{K} N_i r_{ik}\log\left(\frac{r_{ik}}{\eta_{ik}}\right) + \xi_i\left(\sum_{k=1}^{K} r_{ik} - 1\right) \quad (23)$$

Let the partial derivative of $f(r_{i1}, r_{i2}, \ldots, r_{iK})$ to r_{ik} be 0,

$$\begin{aligned} &\frac{\partial f(r_{i1}, r_{i2}, \ldots, r_{iK})}{\partial r_{ik}} = 0 \\ &\Rightarrow D_{ik} + \lambda N_i\left[\log\left(\frac{r_{ik}}{\eta_{ik}}\right) + r_{ik}\frac{\eta_{ik}}{r_{ik}}\frac{1}{\eta_{ik}}\right] + \xi_i = 0 \end{aligned} \quad (24)$$

$$\Rightarrow r_{ik} = \exp\left(-\frac{\xi_i}{\lambda N_i} + 1\right)\eta_{ik}\exp\left[-\frac{D_{ik}}{\lambda N_i}\right] \tag{25}$$

As

$$\sum_{k'=1}^{K} r_{ik'} = 1 \Rightarrow \sum_{k'=1}^{K} \eta_{ik'}\exp\left[-\frac{D_{ik'}}{\lambda N_i}\right]\exp\left(-\frac{\xi_i}{\lambda N_i} + 1\right) = 1 \tag{26}$$

$$\Rightarrow 1 = \exp\left(-\frac{\xi_i}{\lambda N_i} + 1\right)\sum_{k'=1}^{K} \eta_{ik'}\exp\left[-\frac{D_{ik'}}{\lambda N_i}\right] \tag{27}$$

Equation (28) was obtained by taking Equation (25) divided by Equation (27),

$$r_{ik} = \frac{\eta_{ik}\exp\left(-\frac{1}{\lambda N_i}D_{ik}\right)}{\sum\limits_{k'=1}^{K} \eta_{ik'}\exp\left(-\frac{1}{\lambda N_i}D_{ik'}\right)} \tag{28}$$

Equation (29) was obtained by substituting Equation (18) into Equation (28),

$$r_{ik} = \frac{\eta_{ik}\exp\left(\frac{1}{2\lambda}\log|\boldsymbol{\Sigma}_k| + \frac{1}{2\lambda N_i}\sum\limits_{a=1}^{N_i}(z_a - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(z_a - \boldsymbol{\mu}_k)\right)}{\sum\limits_{k'=1}^{K}\eta_{ik'}\exp\left(\frac{1}{2\lambda}\log|\boldsymbol{\Sigma}_{k'}| + \frac{1}{2\lambda N_i}\sum\limits_{a=1}^{N_i}(z_a - \boldsymbol{\mu}_{k'})^T\boldsymbol{\Sigma}_{k'}^{-1}(z_a - \boldsymbol{\mu}_{k'})\right)} \tag{29}$$

2.2.2. Parallel RHMRF-FCM Segmentation

When the number of pixels is huge, or the image information is complex, the number of regions generated will also be very large, which leads to protracted runtime of the serial RHMRF-FCM algorithm. In addition, for multicore parallel computing systems, the communication time between cores is extremely expensive. Effectively reducing the amount of data synchronization can significantly improve the performance of parallel algorithms [42]. Therefore, a parallel RHMRF-FCM algorithm was proposed based on a master-slave parallel mode. The algorithm is described in detail as follows.

For a parallel computing system consisting of P parallel computing units and a coordinated computing unit, in order to make full use of computing performance brought by parallel computing, the dataset $\boldsymbol{Z}$ was equally partitioned into P subsets according to the number of regions, such that $\boldsymbol{Z} = \{\boldsymbol{Z}_l; l = 1, 2, \ldots, p\}$. $\boldsymbol{Z}_l = \{\boldsymbol{Z}_{lq}; q = 1, 2, \ldots, N_l\}$ is the l-th subset data, $\boldsymbol{Z}_{lq} = \{z_{lqa}; a = 1, 2, \ldots, N_q\}$ is the data of region q in the subset$l$, a is the pixel index of region q, and z_{lqa} is the spectral measure of pixel a in the region q of subset$l$. N_l is the number of regions in subset l, which can be computed as:

$$N_l = \begin{cases} \left\lceil \frac{I}{p} \right\rceil, l = 1, 2, \ldots, p-1 \\ I - (p-1)\left\lceil \frac{I}{p} \right\rceil, l = p \end{cases} \tag{30}$$

Similarly, other local variables in the calculation process were defined according to the same rules. The partition of neighborhood regions index set is $\boldsymbol{U} = \{\boldsymbol{U}_l; l = 1, 2, \ldots, p\}$, where $\boldsymbol{U}_l = \{NF_{lq}; q = 1, 2, \ldots, N_l\}$ is the l-th subset of neighborhood regions index set. The partition of prior distribution region labels is $\boldsymbol{\eta} = \{\boldsymbol{\eta}_l; l = 1, 2, \ldots, p\}$, where the l-th subset is $\boldsymbol{\eta}_l = \{\boldsymbol{\eta}_{lq} = \{\eta_{lqk}; k = 1, 2, \ldots, K; q = 1, 2, \ldots, N_l\}$. The partition of region label set is $\boldsymbol{L} = \{\boldsymbol{L}_l; l = 1, 2, \ldots, p\}$, where the l-th subset is $\boldsymbol{L}_l = \{\boldsymbol{L}_{lq} \in \{1, 2, \ldots, K\}; q = 1, 2, \ldots, N_l\}$. $\boldsymbol{M} = \{\boldsymbol{M}_l = \{M_{lk}; k = 1, 2, \ldots, K\}; l = 1, 2, \ldots, p\}$ is a temporary variable introduced to facilitate parallel computing, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_l; l = 1, 2, \ldots, p\}$ is a clustering parameter, the l-th subset is $\boldsymbol{\theta}_l = \{(\boldsymbol{\mu}_{lk}, \sum_{lk}); k = 1, 2, \ldots, K\}$, and $\boldsymbol{\mu}_{lk}$ and $\sum_{lk}$ are the mean vector and covariance matrix of class k in the l-th subset, respectively. The partition of region fuzzy membership set is $\boldsymbol{R} = \{\boldsymbol{R}_l; l = 1, 2, \ldots, p\}$, given the l-th subset is $\boldsymbol{R}_l = \{\boldsymbol{R}_{lq} = \{r_{lqk}; k = 1, 2, \ldots, K\}; q = 1, 2, \ldots, N_l\}$.

By the master-slave parallel mode, the flow of the parallel RHMRF-FCM algorithm can be summarized as follows:

(1) Initialization algorithm parameters include the iteration index $d = 0$, clustering number K, fuzzy degree λ, and iteration threshold T. Initiate the p parallel computing units and the coordinated computing unit. The p parallel computing units are responsible for the calculation of p subsets of corresponding data sets, while the coordination computing unit is responsible for the dataset partitioning and the allocation of the local parameters.

(2) Coordinate computing unit for data partition and data transmission include the partition data set $\boldsymbol{Z}_l = \{\boldsymbol{Z}_{lq}; q = 1, 2, \ldots, N_l\}$. Randomly initialize and partition the region fuzzy membership set $\boldsymbol{R}^{(d)} = \{\boldsymbol{R}_l^{(d)}; l = 1, 2, \ldots, p\}$. Compute and divide the neighborhood region index set $\boldsymbol{U} = \{\boldsymbol{U}_l; l = 1, 2, \ldots, p\}$, then transmit the partitioned data sets $\boldsymbol{Z}_l$, $\boldsymbol{R}_l^{(d)}$ and $\boldsymbol{U}_l$ to the corresponding threads.

(3) Compute, in parallel, each label field subset $\boldsymbol{L}_l^{(d)} = \{\boldsymbol{L}_{lq}^{(d)}; q = 1, 2, \ldots, N_l\}$, prior distribution $\boldsymbol{\eta}_l^{(d)} = \{\boldsymbol{\eta}_{lq}^{(d)} = \{\eta_{lqk}^{(d)}; k = 1, 2, \ldots, K\}\}$, $\boldsymbol{M}_l^{(d)} = \{M_{lk}^{(d)}; k = 1, 2, \ldots, K\}$, and $\boldsymbol{\mu}_l^{(d)} = \{\boldsymbol{\mu}_{lk}^{(d)}; k = 1, 2, \ldots, K\}$ on p computation units. Calculate $\boldsymbol{L}_{lq}^{(d)}$, $\eta_{lqk}^{(d)}$, $M_{lk}^{(d)}$, and $\boldsymbol{\mu}_{lk}^{(d)}$ in parallel threads successively by executing the following equations:

$$\boldsymbol{L}_{lq}^{(d)} = \underset{k}{\operatorname{argmax}}\left\{r_{lqk}^{(d)}; k = 1, 2, \ldots, K\right\} \tag{31}$$

$$\eta_{lqk}^{(d)} = \frac{\exp\left(-\beta \sum\limits_{\boldsymbol{F}_{q'} \in NF_{lq}} t\left(\boldsymbol{L}_{lq}^{(d)} = k, \boldsymbol{L}_{q'}^{(d)}\right)\right)}{\sum\limits_{k'=1}^{K} \exp\left(-\beta \sum\limits_{\boldsymbol{F}_{q'} \in NF_{lq}} t\left(\boldsymbol{L}_{lq}^{(d)} = k', \boldsymbol{L}_{q'}^{(d)}\right)\right)} \tag{32}$$

$$M_{lk}^{(d)} = \sum_{q=1}^{N_l} N_q r_{lqk}^{(d)} \tag{33}$$

$$\boldsymbol{\mu}_{lk}^{(d)} = \sum_{q=1}^{N_l} r_{lqk}^{(d)} \sum_{a=1}^{N_q} z_{lqa} \tag{34}$$

(4) Reduce $\boldsymbol{M}_l^{(d)}$ and $\boldsymbol{\mu}_l^{(d)}$ using the coordinate computing unit from p computation units, and calculate $M_k^{(d)}$ and $\boldsymbol{\mu}_k^{(d)}$ using Equations (35) and (36). Then, synchronize the parameters$M_k^{(d)}$ and $\boldsymbol{\mu}_k^{(d)}$ to p parallel computing units.

$$M_k^{(d)} = \sum_{l=1}^{p} M_{lk}^{(d)} \tag{35}$$

$$\boldsymbol{\mu}_k^{(d)} = \frac{\sum\limits_{l=1}^{q} \boldsymbol{\mu}_{lk}^{(d)}}{M_k^{(d)}} \tag{36}$$

(5) Compute $\sum_l^{(d)} = \{\sum_{lk}^{(d)}; k = 1, 2, \ldots, K\}$ using p parallel computing units by:

$$\boldsymbol{\Sigma}_{lk}^{(d)} = \frac{\sum\limits_{q=1}^{N_l} r_{lqk}^{(d)} \sum\limits_{a=1}^{N_q} \left(z_{lqa} - \boldsymbol{\mu}_k^{(d)}\right)^T \left(z_{lqa} - \boldsymbol{\mu}_k^{(d)}\right)}{M_k^{(d)}} \tag{37}$$

(6) Reduce $\sum_l{}^{(d)}$ from p computation units, and calculate the $\sum_k{}^{(d)}$ by Equation (38). Then, synchronize the $\sum_k{}^{(d)}$ to p parallel computational units:

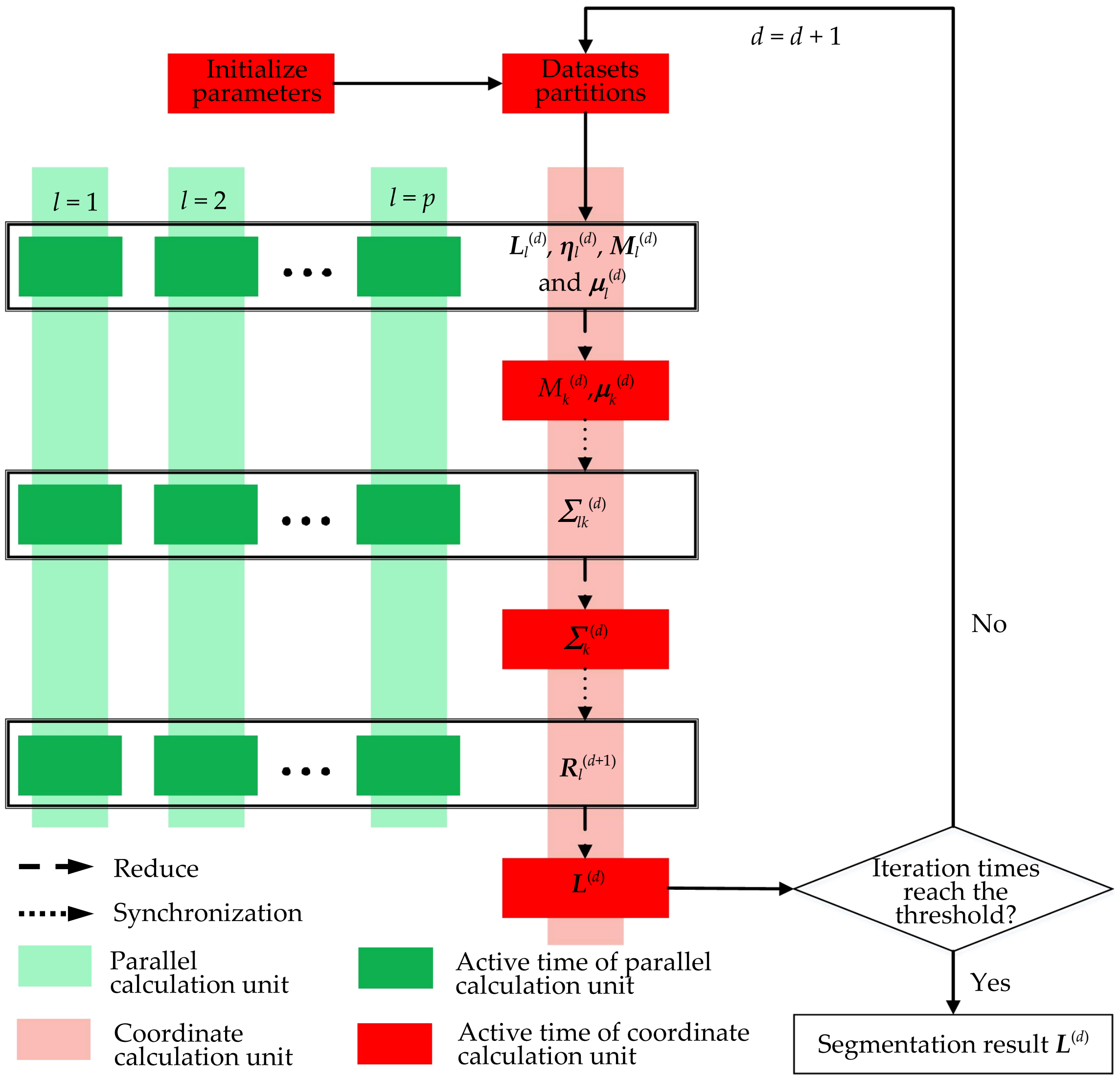
$$\boldsymbol{\Sigma}_k^{(d)} = \sum_{l=1}^{q} \boldsymbol{\Sigma}_{lk}^{(d)} \tag{38}$$

(7) Compute $\boldsymbol{R}_l{}^{(d+1)} = \{\boldsymbol{R}_{lq}{}^{(d+1)} = \{r_{lqk}{}^{(d+1)}; k = 1, 2, \ldots, K\}; q = 1, 2, \ldots, N_l\}$ in p parallel computing units using the equation:

$$r_{lqk}^{(d+1)} = \frac{\eta_{lqk}^{(d)} \exp\left(\frac{1}{2\lambda} \log\left|\boldsymbol{\Sigma}_k^{(d)}\right| + \frac{1}{2\lambda N_q} \sum_{a=1}^{N_q} \left(z_{lqa} - \boldsymbol{\mu}_k^{(d)}\right)^T \left(\boldsymbol{\Sigma}_k^{(d)}\right)^{-1} \left(z_{lqa} - \boldsymbol{\mu}_k^{(d)}\right)\right)}{\sum_{k'=1}^{K} \eta_{lqk'}^{(d)} \exp\left(\frac{1}{2\lambda} \log\left|\boldsymbol{\Sigma}_{k'}^{(d)}\right| + \frac{1}{2\lambda N_q} \sum_{a=1}^{N_q} \left(z_{lqa} - \boldsymbol{\mu}_{k'}^{(d)}\right)^T \left(\boldsymbol{\Sigma}_{k'}^{(d)}\right)^{-1} \left(z_{lqa} - \boldsymbol{\mu}_{k'}^{(d)}\right)\right)} \tag{39}$$

(8) If the iterations threshold is reached, the partition result $\boldsymbol{L}^{(d)}$ is reduced and the loop is withdrawn. Otherwise, $d = d$+1, and then return to step (3) in the iteration process.

In order to more clearly define the local parallel computation, the parallel RHMRF-FCM algorithm was reduced and synchronized. The flowchart of the algorithm is shown in Figure 13. The local variables in each parallel computing unit were $\boldsymbol{Z}_l$, $\boldsymbol{U}_l$, $\boldsymbol{R}_l$, $\boldsymbol{L}_l$, $\boldsymbol{\eta}_l$, $\boldsymbol{M}_l$, and $\boldsymbol{\theta}_l$. The variables stored in the computation procedure of the coordinated computing unit were M_k, $\boldsymbol{\mu}_k$, and $\sum_k$. As shown in Figure 13, the parallel computing units were responsible for the calculation of parameters related to data and the main computing carriers of the algorithm. The coordination computing unit was responsible for the generation of local parameters, the calculation of global parameters, and the synchronization of global parameters to each parallel computing unit. In addition, the parallel RHMRF-FCM algorithm only needed to specify and synchronize the global model parameters, which mainly included $\boldsymbol{M}_l{}^{(d)}$, $\boldsymbol{\mu}_l{}^{(d)}$, and $\sum_l{}^{(d)}$. The synchronization parameters mainly included $M_k{}^{(d)}$, $\boldsymbol{\mu}_k{}^{(d)}$, and $\sum_k{}^{(d)}$. Based on the definition of the parameters, the amount of data of these parameters was only linearly related to the number of parallel computing units p, resulting in the algorithm being a parallel scheme with low data traffic.



$d = d + 1$
Initialize parameters
Datasets partitions
$l = 1$
$l = 2$
$l = p$
$L_l^{(d)}$, $\eta_l^{(d)}$, $M_l^{(d)}$ and $\mu_l^{(d)}$
$M_k^{(d)}, \mu_k^{(d)}$
$\Sigma_{lk}^{(d)}$
$\Sigma_k^{(d)}$
$R_l^{(d+1)}$
$L^{(d)}$
No
Iteration times reach the threshold?
Yes
Segmentation result $L^{(d)}$
Reduce
Synchronization
Parallel calculation unit
Active time of parallel calculation unit
Coordinate calculation unit
Active time of coordinate calculation unit

Figure 13. Runtime flowchart of the parallel RHMRF-FCM algorithm.

3. Results and Discussion

Our previous works (serial method), presented in Reference [10], discussed the influence of the parameters of serial MHR partitioning method to the final segmentation result, obtained available range of parameters, and compared the multiresolution and watershed segmentation method of the eCognition software, so we will not be repeatedly discussing them. In this study, we focused on the discussion of the cost of the parallel segmentation method and parallel performance analysis. The cost analysis was mainly concerned with the impact of the proposed parallel segmentation schemes on global segmentation results under different data tessellation scales and was evaluated qualitatively and quantitatively. The parallel performance analysis involved the assessment of runtime and speedup [43] of parallel MHR partitioning, parallel regionalization segmentation, and the proposed segmentation method with different cores.

3.1. Cost Analysis

In order to qualitatively and quantitatively evaluate the influence of block tessellation on the proposed parallel segmentation method, a WorldView-3 Pan-Sharpen multi-band image with 1024×1024 pixels and spatial resolution of 0.5 m was used as the experimental image (see Figure 14a). The image shows a factory in an urban area and contains a large number of peripheral objects, such as cars, landscape vegetation, and traffic signs. Figure 14b presents a standard segmented map, which was interpreted artificially and contains five homogeneous regions marked in red (I), green (II), blue (III), yellow (IV), and pink (V). Figure 13c shows a superimposed image of the boundaries of the standard segmentation result and the original image.

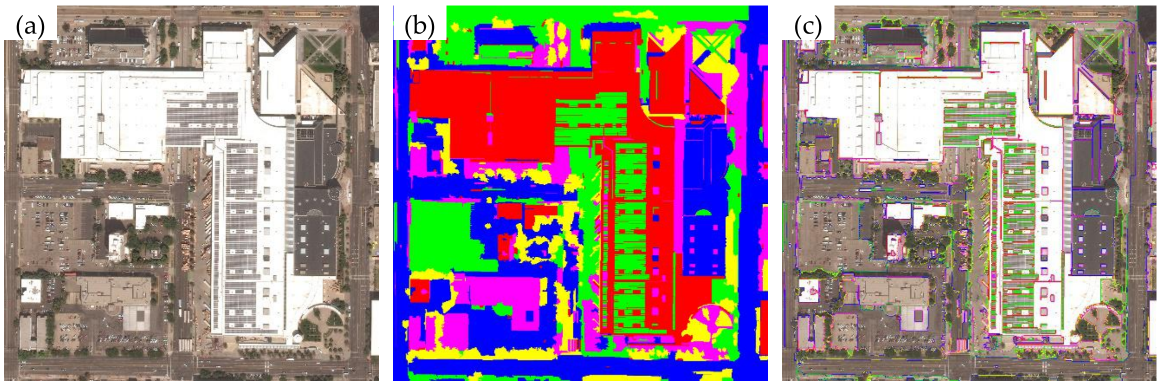


Figure 14. Experiment data and standard segmentation result. (**a**) is an experiment image, (**b**) is the standard segmentation result, and (**c**) is the experiment image superimposed with the regional boundaries of the standard segmentation result.

In order to quantitatively and qualitatively assess the influence of different block tessellation scales on the final segmentation results, four comparative experiments were implemented. For experiments 1–3, the image was tessellated into 64, 16, and 4 blocks with tessellation scales 128×128, 256×256, and 512×512, respectively. In experiment 4, the image was not tessellated. Parallel MHR partitioning technique for the first three experiments, while the fourth setup, which served as the control, was divided using the serial MHR partitioning method. According to Reference [10], the segmentation parameters used in the four experiments were as follows: Dividing scale $S = 220$, the weight of spectral measure similarity $\alpha = 0.9$, the weight of region shape compactness $\gamma = 0.1$, the threshold of iteration number $T = 40$, fuzzy degree $\lambda = 0.1$, homogeneous regions $K = 5$, and the clique potentials of label field $\beta = 0.1$.

The results are presented in Figure 15. Figure 15a–p show the results of experiments 1–4, respectively. Figure 15a,e,i,m show the results using MHR partitioning procedure, with different colors used to identify partitions. The resulting subdividing outlines overlaid over the original images are shown in Figure 15b,f,j,n. Figure 15c,g,k,o present the optimal segmentation results by parallel RHMRF-FCM, and Figure 15d,h,l,p are the optimal segmentation outlines overlaid on the original images.

As shown in Figure 15a,e,i,m, block merging can significantly reduce block tessellation. However, some regions were still affected by high heterogeneity in pixel spectral information, which results in difficulties to effectively process block merging. Areas (e.g., roads and parking lots) containing a substantial amount of peripheral objects (e.g., vehicles, traffic markers, and shadows) can exhibit high levels of spectral heterogeneity. When block tessellation boundaries encounter these types of objects, they may yield unpredictable effects on the merging results. For terrains with low spectral heterogeneity, such as the factory roof in the image, block merging can eliminate the effects of block tessellation. Comparing the MHR partitioning results in Figure 15a,e,i,m, the increase in block tessellation scale resulted in a significant reduction of the influence of block tessellation on segmentation. The result from the first experiment was visibly different from the other three. The effect of block tessellation could not be effectively eliminated in this setup, which was impacted by the very small tessellation scale (128×128). The results from the third experiment were found to be comparable with the control group. The superposition analysis shows that the outlines of Figure 15b,f,j,n matched the complex boundaries of objects well. Generally, the segmentation results were reasonably similar, with slight differences found in local areas. Also, excessive segmentation was found to decrease with the increase in the block tessellation scale.

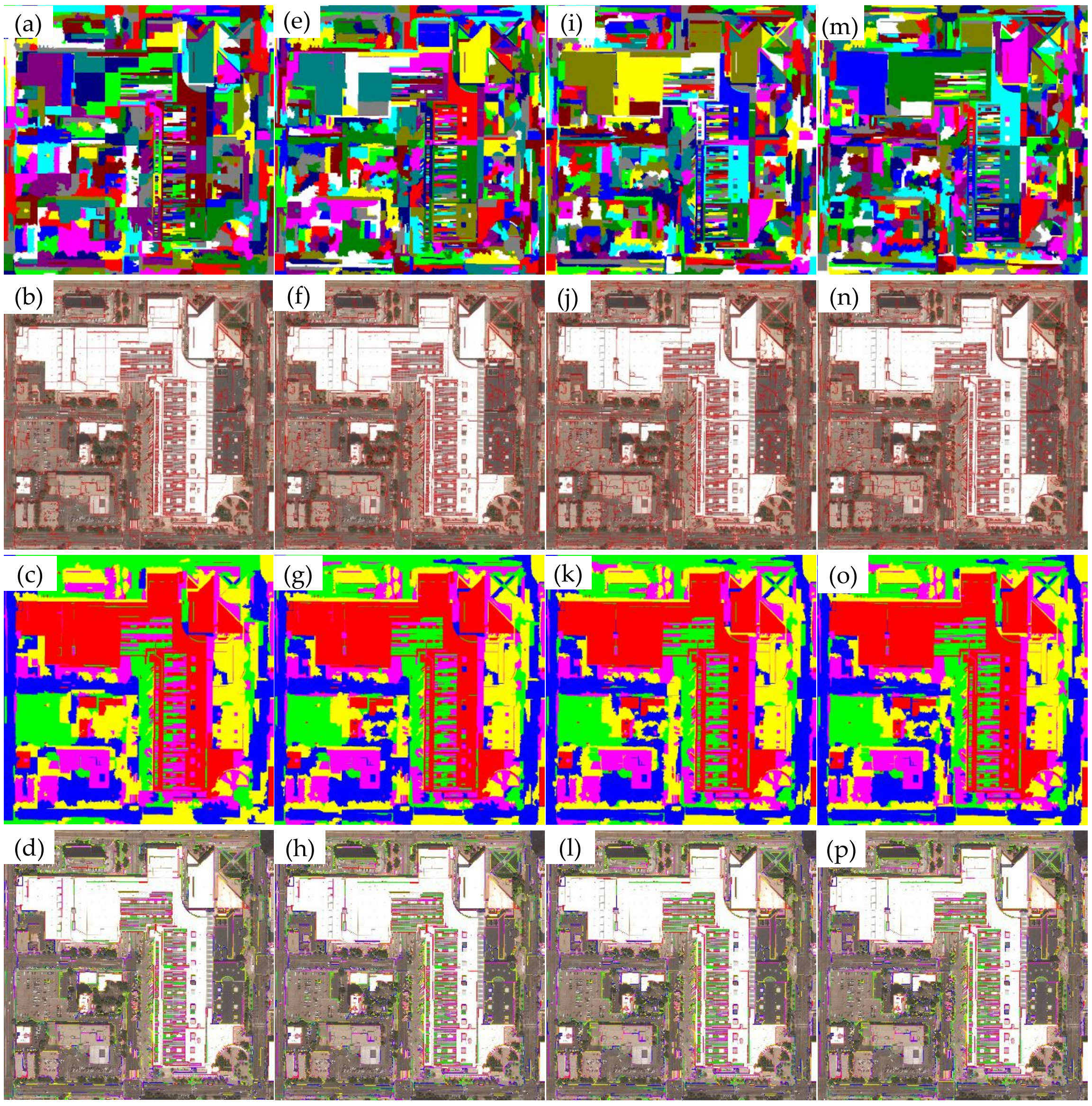


Figure 15. Segmentation result by different tessellation scale.

Comparing the optimal segmentation results in Figure 15c,g,k,o, the four optimal segmentation results with different tessellation scales were generally similar but with slight variations locally. When the tessellation scale was increased, the difference between the local segmentation results and the control group showed a decreasing trend. These results show that the proposed RHMRF-FCM optimal segmentation method can accurately assess the class of homogeneous regions globally and can further reduce the influence of block tessellation. The optimal segmentation results had slight differences locally, mainly due to the influence of block tessellation. Based on the results of the superposition analysis (Figure 15d,h,l,p), the segmentation boundaries are able to delineate irregular boundaries more accurately, can overcome the influence of most peripheral objects, and are able to generate better segmentation results.

In order to quantitatively evaluate segmentation accuracies, the confusion matrix [44] was used in the quantitative segmentation accuracy evaluation model. The manual interpretation result of

Figure 14b was used as the standard segmentation results, and the evaluation results are listed in Table 1.

Table 1. Accuracy evaluation of segment results with different tessellation scale.

Block Size	Accuracy	Homogeneous Region				
		I	II	III	IV	V
128 × 128	User's	0.98	0.69	0.86	0.39	0.86
	Producer's	0.98	0.81	0.57	0.80	0.76
	Overall			0.76		
	Kappa			0.72		
256 × 256	User's	0.99	0.78	0.89	0.44	0.94
	Producer's	0.99	0.88	0.64	0.83	0.82
	Overall			0.82		
	Kappa			0.78		
512 × 512	User's	0.99	0.80	0.93	0.46	0.96
	Producer's	0.99	0.89	0.65	0.90	0.84
	Overall			0.83		
	Kappa			0.80		
1024 × 1024	User's	0.99	0.83	0.94	0.48	0.96
	Producer's	0.99	0.89	0.67	0.92	0.86
	Overall			0.85		
	Kappa			0.82		

The results show that the segmentation accuracies of the first experiment were low, which suggests that the 128 × 128 block tessellation scale had a significant influence on the segmentation accuracy. The overall accuracies in the last three groups were at least 0.82, and the kappa values were at least 0.78. Also, the user's and producer's accuracies for each homogeneous region were mostly above 0.8. The results suggest the total segmentation accuracy was able to reach the standard for an excellent classifier. The overall accuracy and kappa value of the second group were lower than the control group by 0.03 and 0.04, respectively. The overall accuracy and kappa value of the third group were lower than the control group by 0.02 and 0.02, respectively. These suggest that the 256 × 256 and 512 × 512 tessellation scales had less effect on the final segmentation results. With the increasing block tessellation scale, the precision index generally increased in small amounts. In addition, the producer's accuracies in regions III and IV were only 0.64–0.67 and 0.44–0.48, respectively, which were different from the accuracies of other regions. For further comparison, the standard segmentation results and the control group segmentation results were compared, as shown in Figure 16. Figure 16a is the standard segmentation result. The result's outlines were extracted and overlaid on the original image, as shown in Figure 16c. Figure 16b shows the result of the control group segmentation. The result's outlines were extracted and overlaid on the original image, shown in Figure 16d. Homogeneous regions III and IV correspond to the blue and yellow areas in Figure 16a,b, respectively. Compared with Figure 16c, homogeneous area III corresponds to dark pavement and shadows and contains a large number of peripheral features, such as vehicles and traffic signs, while homogeneous area IV corresponds to independent trees, lawns, etc. Visually, the two homogeneous regions show strong heterogeneity in spectral characteristics, and the color difference between them is not distinctly apparent, which exacerbates the difficulty of distinguishing the two homogeneous regions. In order to more accurately reflect the distribution of spectral characteristics of the two homogeneous regions, their pixel indexes were extracted, and their spectral information was obtained from the original map. The spectral distributions of the two homogeneous regions are shown in Figure 17. The red points represent the spectral information of pixels in region III, while the blue points represent the spectral information for pixels in region IV. As seen from the graph, the entanglement of spectral features in the homogeneous regions III and IV was the fundamental reason for their low segmentation accuracy.

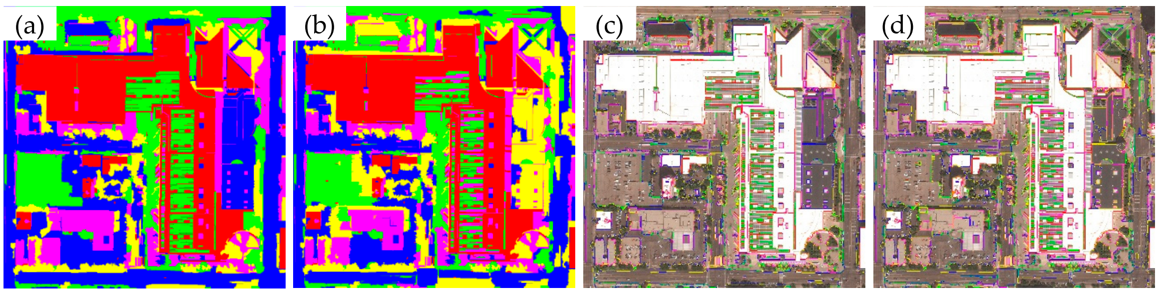


Figure 16. Comparison between the standard and control group segmentation results, (**a**,**b**) are the segmentation results of the standard and control group, respectively, (**c**,**d**) are the superposition image of the standard and control group, respectively.

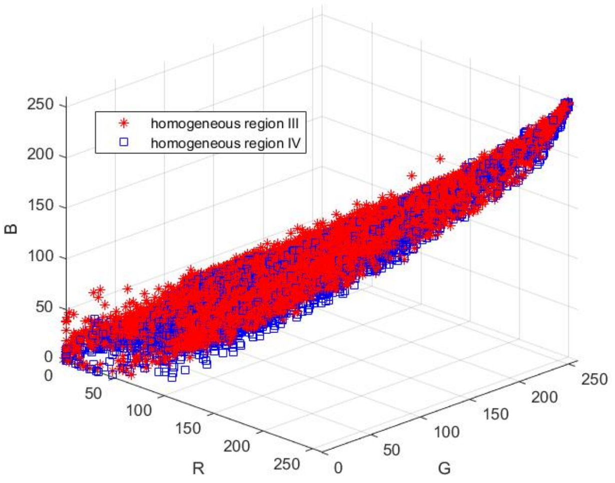


Figure 17. Spectral features distribution of III and IV homogeneous regions.

3.2. Performance Analysis

A Dell Power Edge R930 server with four sockets was used for the parallel performance analysis of the proposed parallel image segmentation method. The server had four Intel Xeon E7–8880 v3 @2.3GHz CPU, and each CPU had 18 cores (72 cores in total), 96 GB, 2133 MT/s memory, and a Samsung EVO Pro 512 GB SSD used as the operating system disk. The parallel computing toolbox of Matlab2018a software running on the Windows 10 operating system was used to implement the proposed method, which provided a parallel way through a multiworker mode.

In order to verify the parallel performance of the proposed parallel image segmentation method, two WorldView-3 Pan-Sharpened multiband images with 8192 × 8192 pixels/0.5-m spatial resolution/urban area of Saltlake, USA, and 7680 × 7680 pixels/0.3-m spatial resolution/Aeropueto de Madrid-Barajas Adolfo Suarez, Madrid, Spain, were used as experimental images (see Figure 18a,e), respectively. The segmentation parameters of the first experiments were set as same as those in Section 3.1, except the homogeneous regions K = 6 and the block tessellation scale 512 × 512. The segmentation parameters of second experiments were set as same as those in Section 3.1, except the homogeneous regions $K = 6$, the block tessellation scale was 512 × 512, the dividing scale $S = 2000$, the weight of spectral measure similarity $\alpha = 0.7$, and the weight of region shape compactness $\gamma = 0.4$. In addition, considering the main land cover type was bare land, which had high heterogeneity in second experiment image (Figure 18e), the parameters (S, α, γ) of first experiment were different than the second experiment. S and γ were increased and α was decreased to improve the performance of the noice immunity of the parallel MHR method.

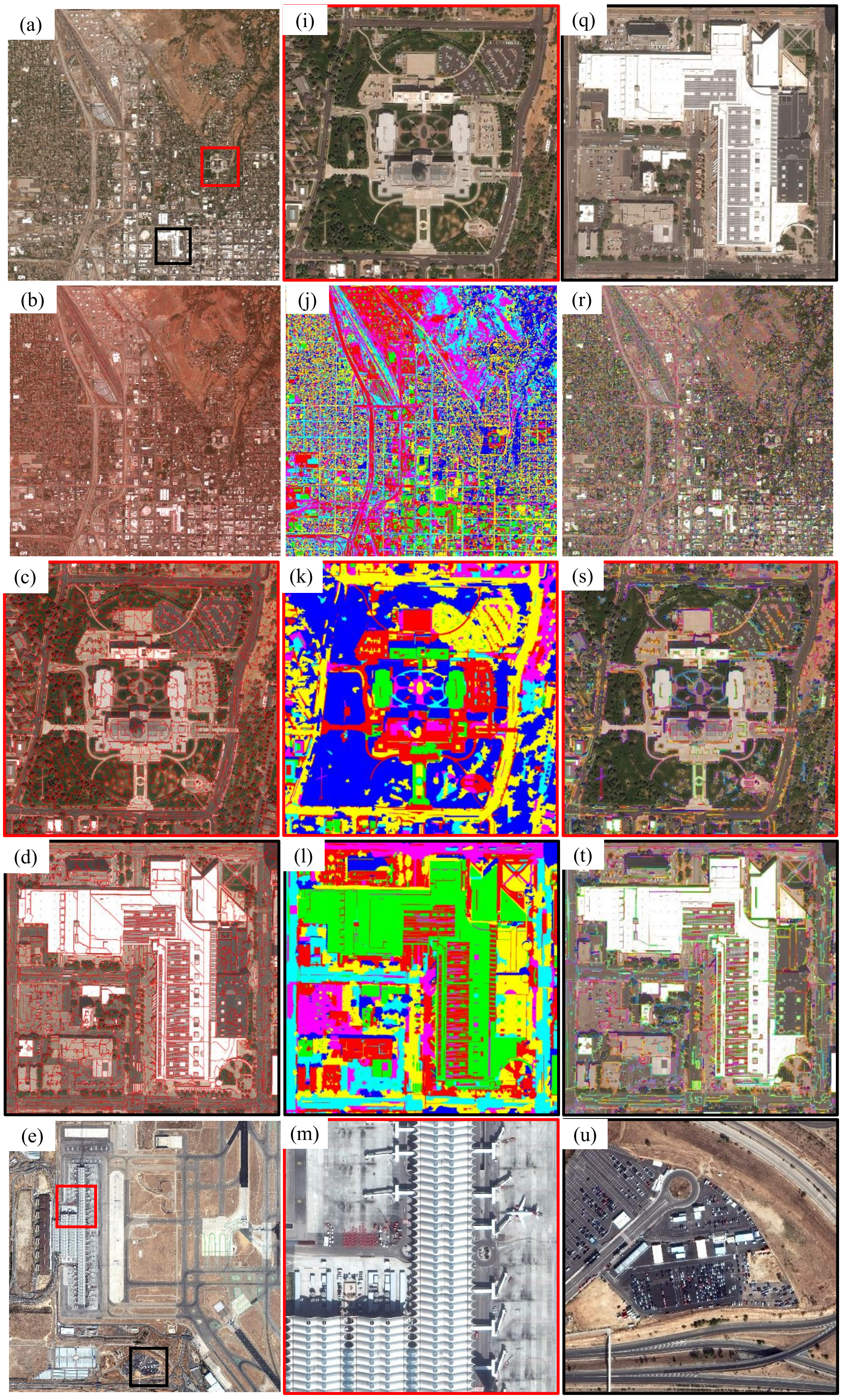


Figure 18. *Cont.*

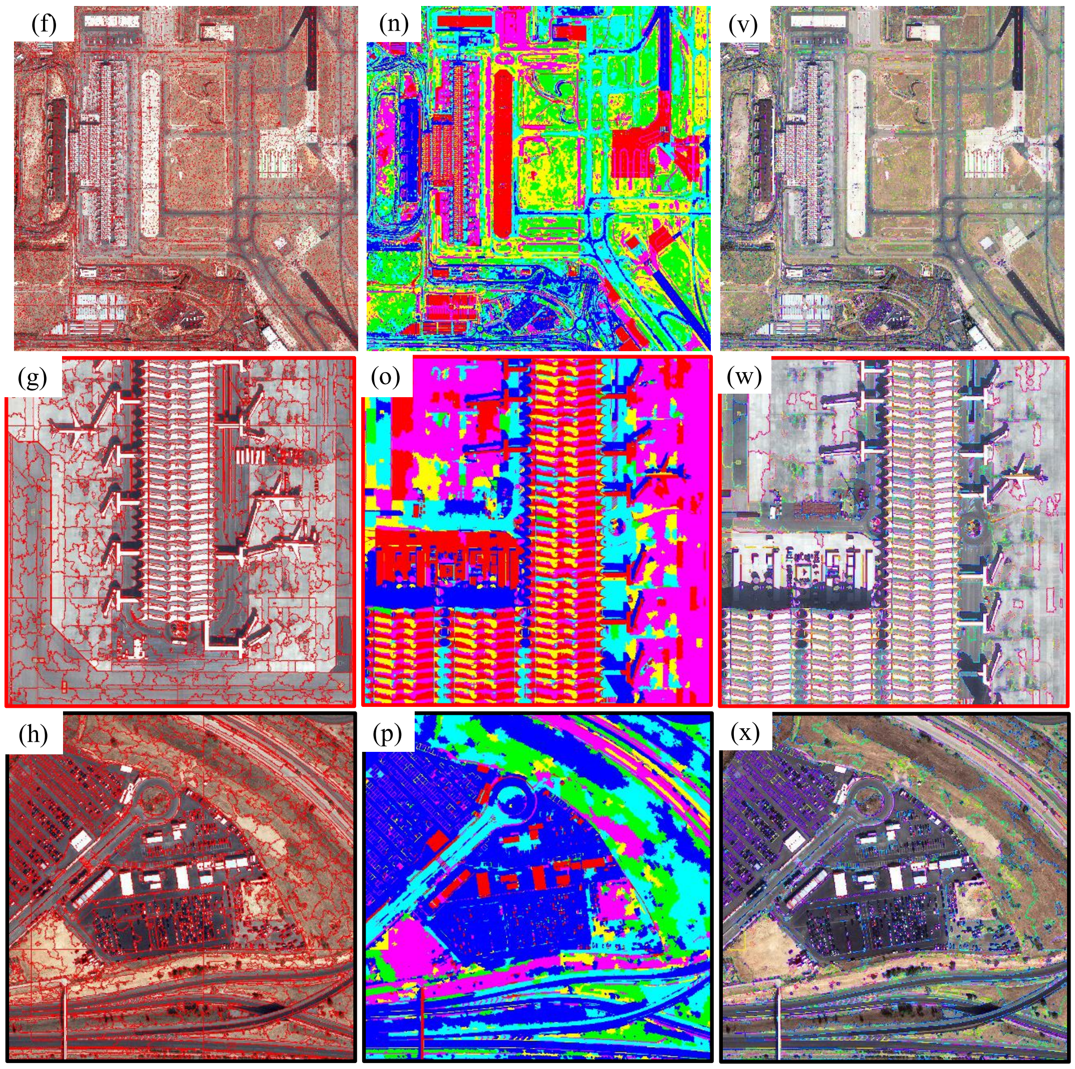


Figure 18. Segmentation results of the whole scene images.

The experimental results are shown in Figure 18. The magnified areas highlighted in red and black boxes in Figure 18a,e are presented in Figure 18i,q (the images of Utah state office buildings and Salt Palace Convention Center) and Figure 18m,u (the images of airport terminal and parking lot), respectively. Figure 18b–d,f–h show the results of the segmentation outlines overlaid on the image, respectively. Figure 18j–l,n–p are the optimal segmentation results, respectively, while Figure 18r–t,v–x are the optimal segmentation outlines overlaid on the image, respectively. The findings suggest that visually, the results obtained by parallel MHR partitioning were relatively good, and the segmentation of complex object boundaries were complete and accurate.

In order to quantitatively evaluate the performance of the proposed parallel segmentation method, the runtime and speedup of the parallel MHR partitioning method, the parallel RHMRF-FCM segmentation method, and the proposed method were compared using different core numbers and segmentation parameters.

The runtime of the parallel MHR partitioning technique was tested using 8, 16, 24, 32, 40, 48, 56, and 64 cores. The results, as illustrated in Figure 19 by a blue curve, show the running time decreased rapidly with the increasing core number until it reached 48. At such point, the runtime became stable. This indicates that the best core number option for this experiment was 48. The speedup in the parallel MHR segmentation method is shown by the red curve in Figure 19. The results show that speedup increased rapidly as the core number increased until it reached 48, wherein the speedup tended to stabilize. The serial runtime of the minimum heterogeneous segmentation of an image with 8192 × 8192 pixels was then estimated by quadratic fitting. The runtime values for 128 × 128, 256 × 256, 512 × 512, 1024 × 1024, and 2048 × 2048 pixel images were used (see Figure 20a). The fitting function and fitting results are presented in Equation (40) and Figure 20b, respectively.

$$f(N) = 4.59 \times 10^{-10} N^2 + 0.02N - 1.04 \times 10^4 \tag{40}$$

where N is the pixel number of the test image. The speedup values may have been overestimated in these experiments. On the one hand, using the theoretical analysis, the serial running time of the 8192 × 8192 pixel image was estimated using the fitting function, which may have resulted in significant deviations from the actual runtime. On the other hand, the time complexity of the serial algorithm in the proposed method was high given that the algorithm uses the MST model, and the calculation efficiency when implementing the model may not have been optimal. However, the speedup curve can also roughly reflect the proposed parallel MHR partitioning method, which had a significantly parallel performance.

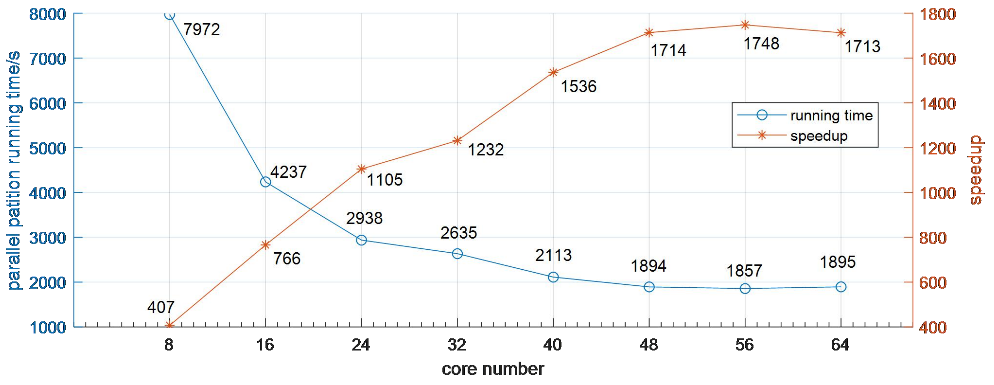


Figure 19. Parallel block partitioning time and its speedup with different core numbers.

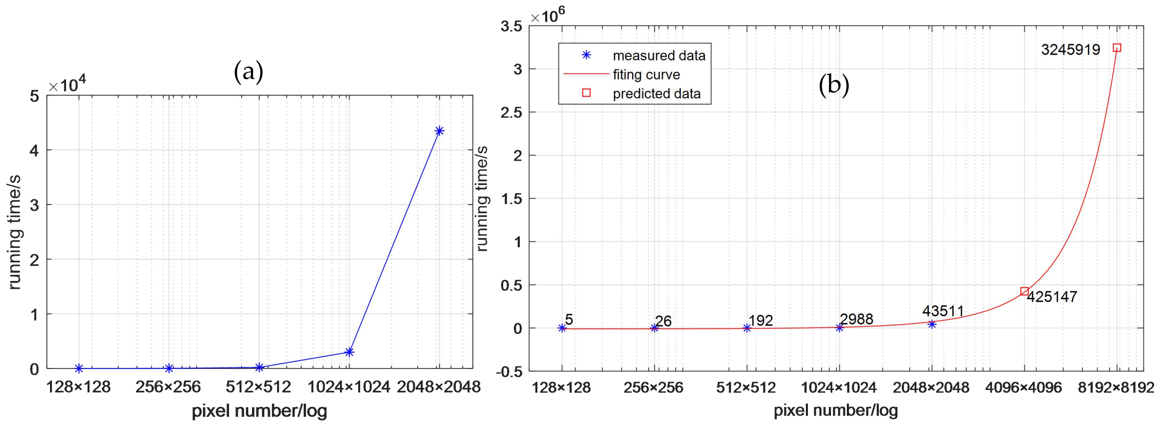


Figure 20. Block partitioning time on different tessellation scale. (**a**) the running time curve of 128 × 128–2048 × 2048 pixel images; (**b**) the fitting running time curve of 128 × 128–8192 × 8192 pixel images.

The runtime of the parallel RHMRF-FCM segmentation method was tested using 1, 2, 4, 8, 16, 24, 32, 40, 48, 56, and 64 cores. The results, presented in a blue curve in Figure 20, show that runtime decreased rapidly with the increasing core number until the core number reached 16, and then the runtime tended to stabilize. When the core number was at 56, the runtime was minimum, which was only 24% of the serial algorithm, and the acceleration effect was significant. Therefore, for the parallel computing environment, using 56 cores for block parallel partitioning was the optimal choice, while 16 was the choice with the high-cost performance. The speedup of the parallel RHMRF-FCM segmentation method is shown by the red curve in Figure 21. As the number of cores increased, the speedup increased rapidly, until the core number reached 16. At such point, the speedup was at 3.5 and became stable, and the effect of parallel acceleration was also significant. In terms of the magnitude of speedup, the parallel optimal segmentation method was two orders lower than the parallel MHR segmentation technique. This is because the serial RHMRF-FCM algorithm was affected by its iterative solution method, and it was difficult to achieve large-scale performance improvements through parallel computing.

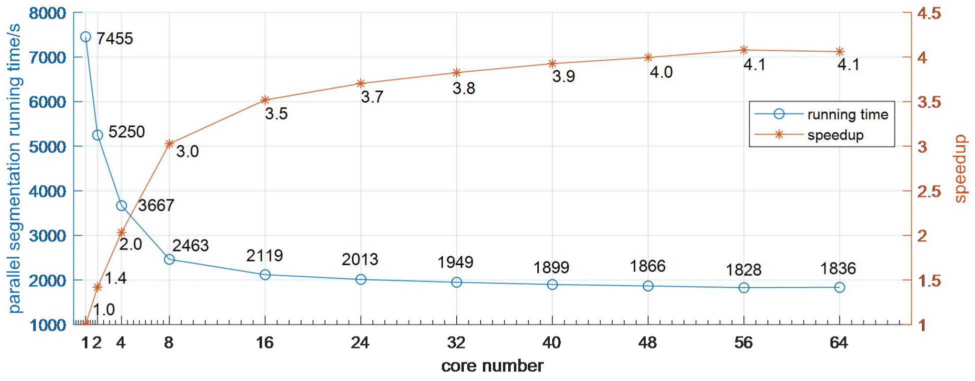


Figure 21. Runtime and speedup of parallel regionalization optimal segmentation method with different core numbers.

Figure 22 shows the runtime (in a blue curve) and speedup (in a red curve) of the proposed parallel segmentation method. The overall trend of the curves shows distinct similarities with the results of the MHR segmentation (Figure 19). With increasing the core number, the runtime and the speedup rapidly decreased and increased, respectively. After 48 cores, the curves tended to stabilize. The best parallel performance was achieved at 56 cores, with a runtime of 3685 s (about 1 h).

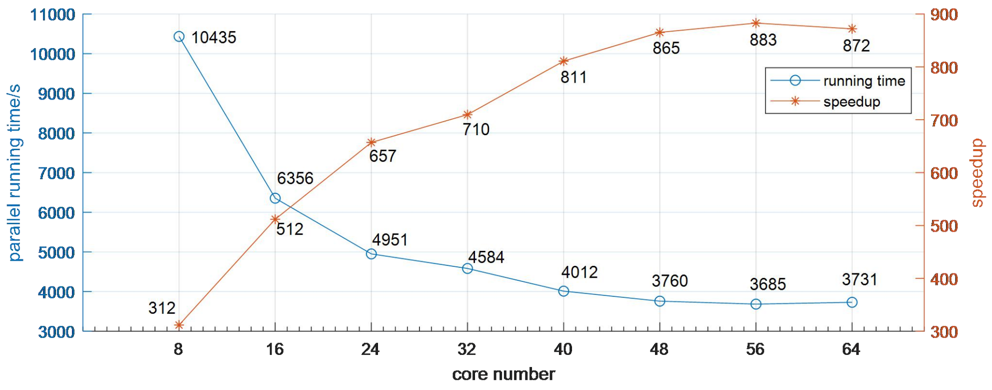


Figure 22. Running time and speedup of the proposed method with different core numbers.

3.3. Discussion

In the parallel cost analysis experiment, we qualitatively and quantitatively evaluated the influence of different block tessellation scale on the final segmentation results. The results suggest the following conclusions: (a) With the increasing block tessellation scale, the similarity increased between the parallel segmentation results and serial segmentation results in terms of visual examination and quantitative accuracy evaluation results. This demonstrates the effectiveness of the proposed parallel MHR partitioning method. (b) For a WorldView-3 remote sensing image with 0.5-m resolution, the most reasonable block tessellation scale was 512 × 512. Using such setting, the accuracy loss in segmentation was constrained within 2% compared with the control group, which is generally acceptable in most applications. In the parallel performance analysis experiment, relatively good results were obtained using the parallel MHR partitioning method. The segmentation results of complex object boundaries were found to be complete and accurate. For a WorldView-3 Pan-Sharpened multiband remote sensing image with 8192×8192 pixels and 0.5-m spatial resolution, the running time and speedup curve for the parallel MHR partitioning, parallel RHMRF-FCM, and the proposed method were examined using four sockets server with 72 Cores. The results demonstrate that the proposed method has a high parallel performance.

4. Conclusions

High-resolution imagery offers rich geometric, texture, and spectral information, and is characterized by big data. These features bring new challenges to traditional image segmentation methods. To this end, this study proposed an efficient parallel high-resolution image segmentation method based on the MST and RHMRF-FCM model. There are two main contributions of this research: (a) On the basis of image domain decomposition by regular tessellation, block partitioning was executed in parallel by the MHR technique and integrated with the block IMSTF model, which effectively expressed the image's explicit and implicit features. The parallel block merging method was then proposed to decrease the impact of landscape caused by image tessellation. (b) A parallel RHMRF-FCM segmentation method based on master-slave parallel mode was proposed to improve the computing performance of the serial RHMRF-FCM method, which effectively decreased the amount of data synchronization in parallel algorithms.

The proposed method has the following advantages: (a) By introducing the shape proportionality expressed on the IMSTF model, it serves as a much improved version of the MHR technique that is able to solve the problems of geometric noise and increased heterogeneity in homogeneous regions of high-resolution images. (b) The new method provides an efficient segmentation framework for big-scale high-resolution images using a parallel MHR partitioning technique, a parallel merging method based on image tessellation, and a parallel RHMRF-FCM segmentation method with low data synchronization traffic parallel scheme.

Further research can focus on the following directions: (a) Other parallel partitioning methods for big-scale high-resolution images can be explored through the creation of a more effective coordination mechanism between blocks. (b) Also, the proposed method can be applied to examine different types of remote sensing data, such as hyperspectral image and SAR image, and other applications, such as feature extraction and target identification.

Author Contributions: This research mainly performed and prepared by W.L. and Y.L. W.L. contributed with ideas, conceived, and designed the study. Y.L. supervised the study and his comments were considered throughout the paper. All authors have read and agreed to the published version of the manuscript.

Funding: The research was funded by: (1) the National Natural Science Foundation of China (Project No. 41271435); (2) the National Natural Science Foundation of China Youth Science Foundation (Project No. 41301479)

Acknowledgments: Thanks to the anonymous reviewers for their constructive and valuable suggestions on the earlier drafts of this manuscript, and special thanks to the professional English editing service from EditX.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Jensen, J. *Introductory Digital Image Processing: A Remote Sensing Perspective*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2005.
2. Zhao, X.; Li, Y.; Zhao, Q. Mahalanobis distance based on fuzzy clustering algorithm for image segmentation. *Digit. Signal Process.* **2015**, *43*, 8–16. [CrossRef]
3. Li, Y. Remotely Sensed Data Segmentation under a Spatial Statistics Framework. Ph.D. Thesis, University of Waterloo, Waterloo, Ontario, 2009.
4. Curran, P. Remote sensing: Using the spatial domain. *Environ. Ecol. Stat.* **2001**, *8*, 331–344. [CrossRef]

5. Wang, C.; Xu, A.; Li, X. Supervised classification high-resolution remote-sensing image based on interval type-2 fuzzy membership function. *Remote Sens.* **2018**, *10*, 710. [CrossRef]
6. Li, D.; Tong, Q.; Li, R.; Gong, J.Y.; Zhnag, L. Current issues in high-resolution earth observation technology. *Sci. China Earth Sci.* **2012**, *55*, 1043–1051. (In Chinese) [CrossRef]
7. Benediktsson, J.; Chanussot, J.; Moon, W. Very high-resolution remote sensing: Challenges and opportunities. *Proc. IEEE* **2012**, *100*, 1907–1910. [CrossRef]
8. Ma, Y.; Wu, H.; Wang, L.; Huang, B.; Rangjan, R.; Zomaya, A.; Jie, W. Remote sensing big data computing: Challenges and opportunities. *Future Gener. Comput. Syst.* **2015**, *51*, 47–60. [CrossRef]
9. Rathore, M.; Paul, A.; Ahmad, A.; Chen, B.W.; Huang, B.; Ji, W. Real-time big data analytical architecture for remote sensing application. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4610–4621. [CrossRef]
10. Lin, W.; Li, Y.; Zhao, Q. High-resolution remote sensing image segmentation using minimum spanning tree tessellation and RHMRF-FCM algorithm. *Acta Geod. Cartogr. Sin.* **2019**, *48*, 64–74. (In Chinese)
11. Dearo, G.; Coelho, N. Multiple parallel mapreduce k-means clustering with validation and selection. In Proceedings of the 2014 Brazilian Conference on Intelligent Systems, IEEE Computer Society, Sao Paulo, Brazil, 18–22 October 2014; pp. 432–437.
12. Gu, H.; Han, Y.; Yang, Y.; Li, H.; Liu, Z.; Soergel, U.; Blaschke, T.; Cui, S. An efficient parallel multi-scale segmentation method for remote sensing imagery. *Remote Sens.* **2018**, *10*, 590. [CrossRef]
13. Du, Z.; Gu, Y.; Zhang, C.; Zhang, F.; Liu, R.; Sequeira, J.; Li, W. ParSymG: A parallel clustering approach for unsupervised classification of remotely sensed imagery. *Int. J. Digit. Earth* **2016**, 1–19. [CrossRef]
14. Guan, Q.; Clarke, K. A general-purpose parallel raster processing programming library test application using a geographic cellular automata model. *Int. J. Geogr. Inf. Sci.* **2010**, *24*, 695–722. [CrossRef]
15. Xing, J.; Sieber, R.; Kalacska, M. The challenges of image segmentation in big remotely sensed imagery data. *Ann. GIS* **2014**, *20*, 233–244. [CrossRef]
16. Peng, B.; Zhang, L.; Zhang, D. A survey of graph theoretical approaches to image segmentation. *Pattern Recognit.* **2013**, *46*, 1020–1038. [CrossRef]
17. Geetha, M.; Rakendu, R. An improved method for segmentation of point cloud using minimum spanning tree. In Proceedings of the 2014 International Conference on Communications & Signal Processing, Melmaruvathur, India, 3–5 April 2014; pp. 833–837.
18. Sharifi, M.; Kiani, K.; Kheirkhahan, M. A Graph-Based Image Segmentation Approach for Image Classification and Its Application on SAR Images. *Prz. Elektrotech.* **2013**, *89*, 202–205.
19. Wu, Z.; Leahy, R. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1993**, *15*, 1101–1113. [CrossRef]
20. Banerjee, B.; Varma, S.; Buddhiraju, K.; Eeti, L.N. Unsupervised multi-spectral satellite image segmentation combining modified mean-shift and a new minimum spanning tree based clustering technique. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 888–894. [CrossRef]
21. Cui, W.; Zhang, Y. An effective graph-based hierarchy image segmentation. *Intell. Autom. Soft Comput.* **2011**, *17*, 969–981. [CrossRef]
22. Baatz, M.; Schäpe, A. Multiresolution segmentaion: An optimization approach for high quality multi-scale image segmentation. In *Angewandte Geographische Informationsverarbeitung XII*; Strobl, J., Blaschke, T., Griesebner, G., Eds.; Wichmann: Heidelberg, Germany, 2000; pp. 12–23.
23. Judah, A.; Hu, B.X.; Wang, J.G. An Algorithm for Boundary Adjustment toward Multi-Scale AdaptiveSegmentation of Remotely Sensed Imagery. *Remote Sens.* **2014**, *6*, 3583–3610. [CrossRef]
24. Wang, P.; Wei, Z.; Cui, W.; Lin, Z. A image segmentation method based on statistics learning theory and minimum spanning tree. *Geomat. Inf. Sci. Wuhan Univ.* **2017**, *42*, 877–883. (In Chinese)
25. Zhao, Q.; Li, X.; Zhao, X.; Li, Y. Fuzzy ISODATA Image Segmentation Intergrating Voronoi Tessellation HMRF Model. *J. Signal Process.* **2016**, *32*, 1233–1243.
26. Zhao, Q.; Li, X.; Zhao, X.; Li, Y. Remote Sensing Image Segmentation Algorithm with Regional Fuzzy Cluster and Mahalanobis Distance. *J. China Univ. Min. Technol.* **2017**, *46*, 222–228.
27. Lucarini, V. Symmetry-Break in Voronoi Tessellations. *Symmetry* **2009**, *1*, 21–54. [CrossRef]
28. Schneider, R. Weighted faces of Poisson Hyperplane Tessellations. *Adv. Appl. Probab.* **2009**, *41*, 682–694. [CrossRef]
29. Wang, Y.; Li, Y.; Zhao, Q. SAR Image Segmentation Combined Regular Tessellation and M-H Algorithm. *Geomat. Inf. Sci. Wuhan Univ.* **2016**, *41*, 1491–1497.

30. Achanta, R.; Shaji, A.; Smith, K. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2274–2282. [CrossRef] [PubMed]
31. Baatz, M.; Schäpe, A. An optimization approach for high quality multi-scale image segmentation. In *Beiträge Zum AGIT-Symposium*; Herber Wichmann-Verlag: Berlin, Germany, 2000; pp. 12–23.
32. Li, H.; Tang, Y.; Liu, Q.; Ding, H.; Jing, L. An improved algorithm based on minimum spanning tree for multi-scale segmentation of remote sensing imagery. *Acta Geod. Cartogr. Sin.* **2015**, *44*, 791–796. (In Chinese)
33. Zhao, Q.; Li, X.; Li, Y.; Zhao, X. A fuzzy clustering image segmentation algorithm based on Hidden Markov Random Field models and Voronoi Tessellation. *Pattern Recognit. Lett.* **2017**, *85*, 49–55. [CrossRef]
34. Cannon, R.L.; Dave, J.V.; Bezdek, J.C.; Trivedi, M.M. Segmentation of a thematic mapper image using the fuzzy c-means clustering algorithm. *IEEE Trans. Geosci. Remote Sens.* **1986**, *24*, 400–408. [CrossRef]
35. Miyamoto, S.; Ichihashi, H.; Honda, K. *Algorithms for Fuzzy Clustering*; Springer: Heidelberg, Germany, 2008.
36. Prim, R. Shortest connection networks and some generalizations. *Bell Syst. Tech. J.* **1957**, *36*, 1389–1401. [CrossRef]
37. Kruskal, J. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Am. Math. Soc.* **1956**, *7*, 48–50. [CrossRef]
38. Kruskal's Minimum Spanning Tree Algorithm|Greedy Algo-2. Available online: https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/ (accessed on 20 January 2020).
39. Thomas, H.; Charles, E.; Ronald, L.; Clifford, S. *Introduction to Algorithms*, 3rd ed.; The MIT Press: Cambridge, MA, USA, 2009.
40. Chatzis, S.; Varvarigou, T. A fuzzy clustering approach toward hidden Markov random field models for enhanced spatially constrained image segmentation. *IEEE Trans. Fuzzy Syst.* **2008**, *16*, 1351–1361. [CrossRef]
41. Rosu, R.-G.; Giovannelli, J.-F.; Giremus, A.; Vacar, A. Potts model parameter estimation in Bayesian segmentation of piecewise constant images. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Brisbane, Australia, 19–24 April 2015; pp. 4080–4084.
42. Chen, W.; Ostrouchov, G.; Pugmire, D.; Prabhat, M.; Wehner, M.F. A parallel EM algorithm for model-based clustering applied to the exploration of large spatio-temporal data. *Technometrics* **2013**, *55*, 513–523. [CrossRef]
43. Trobec, R.; Slivnik, B.; Bulić, P.; Robic, B. *Introduction to Parallel Computing: From Algorithms to Programming on State-of-the-Art Platforms*; Springer: Beilin, Germany, 2018.
44. Congalton, R.; Green, K. *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices*; Lewis Publishers: Boca Raton, FL, USA, 1999.