A Scalable Tile-Based Framework for Region-Merging Segmentation

Pierre Lassalle, Jordi Inglada, Julien Michel, *Member, IEEE*, Manuel Grizonnet, and Julien Malik

***Abstract*—Processing large very high-resolution remote sensing images on resource-constrained devices is a challenging task because of the large size of these data sets. For applications such as environmental monitoring or natural resources management, complex algorithms have to be used to extract information from the images. The memory required to store the images and the data structures of such algorithms may be very high (hundreds of gigabytes) and therefore leads to unfeasibility on commonly available computers. Segmentation algorithms constitute an essential step for the extraction of objects of interest in a scene and will be the topic of the investigation in this paper. The objective of the present work is to adapt image segmentation algorithms for large amounts of data. To overcome the memory issue, large images are usually divided into smaller image tiles, which are processed independently. Region-merging algorithms do not cope well with image tiling since artifacts are present on the tile edges in the final result due to the incoherencies of the regions across the tiles. In this paper, we propose a scalable tile-based framework for region-merging algorithms to segment large images, while ensuring identical results, with respect to processing the whole image at once. We introduce the original concept of the stability margin for a tile. It allows ensuring identical results to those obtained if the whole image had been segmented without tiling. Finally, we discuss the benefits of this framework and demonstrate the scalability of this approach by applying it to real large images.**

***Index Terms*—Image processing, image segmentation, image tiling, region merging, scalability.**

I. Introduction

RECENT Earth observation satellites, such as QuickBird, WorldView, GeoEye, and Pléiades, provide very high-resolution (VHR) images, which are useful in applications such as environmental monitoring or natural resources management. The Pléiades satellites provide images with a ground sampling distance of 0.5 m and a spatial coverage of 400 km^2 (40 000 $\times$ 40 000 pixels) allowing for detailed observation of the Earth surface. As a result, a scene contains billions of pixels, which represents a large amount of data to process. Dealing with such quantity of data has become a challenging issue for the remote sensing community because of the limitation of memory available on computers. The classical way to solve this problem is to divide these large images into smaller tiles (rectangular image subsets of the image) and process each one of these tiles independently. This operation is called image tiling. For traditional pixelwise or with fixed-size regular neighborhood image processing algorithms, image tiling is straightforward to apply without introducing artifacts in the results. However, those algorithms consider only spectral information from the pixels since a pixel does not have morphological information. That is why new trends known as object-based image analysis (OBIA) [1], object-based image classification, spatial reasoning [2], [3], and geospatial analysis have recently emerged using segmentation techniques to extract objects of interest in the scene and derive spatial relations between them. Some textural and morphological attributes are then computed from these objects for a subsequent classification. Segmentation quality is therefore essential for a correct characterization of these objects. However, as illustrated in Section II, image tiling does not cope well with most of the segmentation algorithms and particularly for region-merging algorithms. In this paper, the focus is on the use of image tiling operations for region-merging algorithms. The ability to make these algorithms scalable to arbitrary large images, while ensuring identical results, is very appealing.

This paper makes the following original contributions.

- We define the concept of stability margin to constrain a segmentation algorithm to satisfy the properties of stability introduced in [4] and determine a formal expression of the margin. We derive this stability margin for region-merging segmentation algorithms, and we also give the definition of a stable segment when using a tiling scheme (see Section III-A3 and A4).
- Leveraging this stability margin, we propose a scalable tile-based framework for region-merging algorithms for the segmentation of images of arbitrary size. This framework aims at ensuring identical regions to those obtained if the whole image had been segmented without tiling (see Section III-B).

Finally, we present some experiments (see Section IV), which demonstrate the following.

- The expression of the stability margin for our generic region-merging algorithm avoids artifacts on the tile edges, while ensuring identical results.
- The feasibility of the new framework to segment full VHR scenes.

Manuscript received November 14, 2014; revised March 17, 2015; accepted April 6, 2015. Date of publication May 4, 2015; date of current version June 10, 2015. The work of P. Lassalle was supported in part by the French Space Agency (CNES) and in part by Communications & Systems (CS) through a three-year Ph.D. grant.

P. Lassalle is with the Centre d'Etudes Spatiales de la Biosphère (CESBIO), Centre National d'Études Spatiales (CNES), 31401 Toulouse, France.

J. Inglada and J. Michel are with the DCT/SI/CB, Centre National d'Études Spatiales (CNES), 31401 Toulouse, France.

M. Grizonnet is with the Earth Observation, Centre National d'Études Spatiales (CNES), 31401 Toulouse, France.

J. Malik is with CS Systèmes d'Information, 31506 Toulouse, France.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2015.2422848

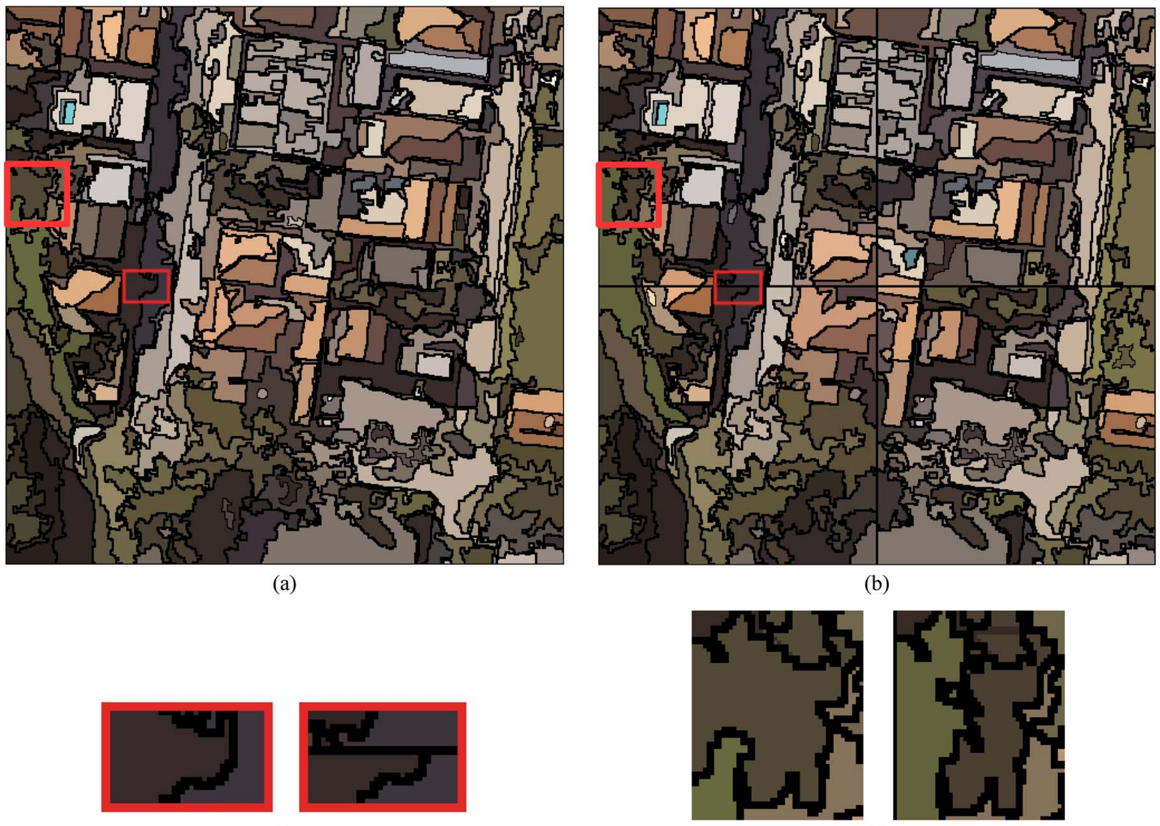


Fig. 1. Impact of image tiling. (a) Represents the output obtained from the reference segmentation of the whole image at once. (b) Represents the output from the tile-based segmentation of the image. The image has been divided into 4 tiles of 250 × 250 pixels. (c) Shows the presence of an artifact on a tile border due to the fact that segments containing pixels along the tile edges are forced to have their contour along the tile edges. (d) Shows a different segment from the test segmentation compared to the one obtained from the reference segmentation due to the absence of knowledge of segments in other tiles. (a) Segmentation without tiling. (b) Tiled segmentation. (c) Example of an artifact. (d) Different segments.

This paper is organized as follows: Section II illustrates the impact of image tiling on the result when performing a region-merging segmentation. An overview of the previous work is presented in Section II-A, and B describes the background elements of region-merging segmentation. Section III gives a description of the proposed solution. Section III-A introduces the concept of the stability margin for segmentation algorithms. It also presents the expression of the stability margin for region-merging algorithms. In Section III-B, we introduce a scalable tile-based framework for region-merging algorithms, which allows the segmentation of images of arbitrary size despite the memory constraints. Finally, Section IV presents some results, which exhibit the correctness of the stability margin expression and the feasibility of the tile-based framework to segment full VHR remote sensing scenes.

II. Problem Statement

In order to illustrate the problem of applying a tiling procedure for image segmentation, we propose the following experiment. A 500 × 500 image is considered. The region-merging algorithm uses the Baatz & Schäpe criterion [5] to form the partition of the image into disjoint segments. This criterion needs three user-defined parameters: two parameters for the relative importance of the spectral and shape weights and a value for the scale threshold. This criterion is described in more detail in Section II-B1. For this experiment, the parameters are set to 0.5, 0.5, and 60, respectively.

The first result is obtained from the segmentation of the whole image at once. This result represents the reference segmentation and is denoted GT. For the second segmentation, the image is first divided into four tiles of 250 × 250 pixels. Each tile is segmented independently, and the result is obtained from the mosaicking of the results of each tile. The result is denoted by TS and represents the tiled segmentation. Fig. 1(a) and (b) shows GT and TS.

The comparison of GT and TS is made by using the Hoover metrics [6]. Four metrics can be used from this comparison. RC is the score of correct segment matches. RF is the score of fragmentation and represents the proportion of segments of GT, which are fragmented into several segments of TS. RA is the score of aggregation and represents the proportion of segments of TS fragmented into several segments of GT. RM is the score of missing detection and represents the proportion of segments of TS and GT, which cannot be classified into the three previous metrics. The scores of the metrics range from 0 to 1. $RC = 1$ means that both segmentations are identical. The scores obtained from the comparison are the following: $RC = 0.8$, $RF = 0.006$, $RA = 0.0325$, and $RM = 0.1018$, reflecting that both segmentations are not identical. Therefore, image tiling has an impact on the final result. Fig. 1(c) shows an example of an artifact on the border of a tile. Indeed, the first consequence of image tiling is that segments located on the tile edges are forced to have their contours along the tile edges. In addition, a discontinuity of the contour can be observed between segments at each side of the tile edge showing a mismatch. Fig. 1(d) shows an example of a segment of TS inside a tile different from the one of GT. Image tiling not only produces artifacts at the tile boundaries but also modifies some merging decisions inside the tiles. These two main consequences have been identified with the stability benchmark described in [4].

A. Previous Work

Several approaches have been investigated to remove the artifacts on the tile edges. In [7] and later in [8], the authors introduced the idea of "contagious" segments. At the beginning of the segmentation procedure, the contagious segments are the pixels along the tile edges. During the merging process, when a segment merges with a contagious segment, the resulting segment becomes contagious. A solution proposed by the authors was to prevent two contagious segments from merging, in order to limit the propagation of the contagious property. However, as mentioned in [9], this approach is unreliable because, oftentimes, there are so many segments, which became contagious, that the region-growing process would stall prematurely. Another idea was to divide the image into adaptative tiles [10]. The borders of the tiles are built along the line of the maximum image gradient. This way, it is expected that the lines follow the border of the segments. However, the authors warn that, in certain cases, this approach creates inconsistent objects. In [9], the authors propose an alternative solution for the contagious segments. They propose the RHSeg algorithm, which is an approximation of the original algorithm HSeg [11]. A split-and-remerge process is performed after each iteration to remerge the contagious segments. This method successfully removes the artifacts on the tile edges but remains an approximation of HSeg. Therefore, the equivalence of the results, with respect to the segmentation without tiling, is not ensured. A different idea was to process the artifacts after stitching the segmented tiles together. In [12], the authors propose using a topological criterion to remove the artifacts on the tile edges. Two segments on each side of a tile border are merged, if their contact surface is greater than a user-defined threshold. This solution does not guarantee the removal of all the artifacts on the tile edges since a region located on one side of the tile edge can have several contact surfaces with segments on the other side of the tile edges. In this case, when this segment is merged with one of the candidate regions, it triggers artifacts located at the contact surfaces between the other segments. More recently, an exact solution for the mean-shift algorithm has been proposed [4] when using image tiling. Unstable operations consisting of optimizing the execution time were removed from the classical mean-shift algorithm. The biconnected component algorithm, which was used in the classical version to assign the labels to the pixels that converged to their modes, was replaced by the connected component algorithm. With this new version, the authors propose dividing the image into tiles containing an additional stability margin. The size of the additional stability margin depends on the spatial range of the mean-shift window and the number of iterations to be performed. They also define a way to stitch the segmented tiles to produce the final output. With this version, they ensure the absence of artifacts on the tile edges and, furthermore, identical results to a segmentation obtained without tiling.

In summary, ensuring identical results when performing image tiling turns out to be a difficult task for region-merging algorithms. Some solutions have been proposed, but they do not ensure the equivalence of the result. However, one solution has been found for the mean shift using the concept of stability margin. In this paper, we extend this approach to any region-merging algorithm.

B. Background Elements of Region-Merging Segmentation

Region-merging algorithms appear to be very well suited for the interpretation of high-resolution images [13] because of their high-quality results compared to other approaches [14]. To obtain a partition of the image, region-based segmentation algorithms [15], [16] do not handle pixels but segments, which are sets of connected pixels. The pixels that belong to the same segment exhibit common properties according to a homogeneity criterion. These algorithms have received a lot of attention from the OBIA community.

Region-merging algorithms start by assigning a different segment to each pixel of the image. The algorithm consists of merging adjacent segments until a termination criterion is fulfilled. At each iteration, merging costs are computed between adjacent segments. These merging costs are based on a homogeneity criterion and can represent not only how two similar segments are but also how homogeneous the resulting larger segment would be. The adjacent segment, for which the merging cost is the smallest compared to the other adjacent segments, is called the best adjacent segment of the given segment. A segment and its best adjacent segment are merged, if their merging cost is smaller than a threshold. This threshold avoids undersegmentation. The merging process stops when there are no more possible fusions of segments. The homogeneity criterion can be based on statistical measures [17], spectral attributes [18], or topological attributes [5]. A specific criterion for partitioning the image can be defined for a particular need. A huge diversity of criteria can therefore exist.

1) Overview of Some Homogeneity Criteria:

- Spectral information

  The first region-growing algorithms, which appear in the literature, were based only on spectral information. A user-defined threshold usually limits the growth of the segments. In [18], the criterion is based on the Euclidean distance between spectral vector values. A segment is described by its spectral mean vector. The similarity between two segments is then the Euclidean distance between their mean vectors. A threshold determines the maximum value of the distance to merge the segments. This region-growing algorithm has been integrated in the SPRING image processing software [19] and recently in the InterImage open-source image processing software

[20]. In [21], the criterion is based on the distance between spectral variance vectors. The region-based segmentation consists of a multiple-pass where adjacent segments with a distance lower than a user-defined global threshold are merged. There is also a minimum segment size, which enforces the construction of segments in areas of high local variance in the image. Finally, in [22], the authors propose an unsupervised region-based segmentation method for hyperspectral data. The first preprocessing step consists of reducing the dimension of the hyperspectral analysis by performing a principal component analysis. Next, the mean-shift filtering is performed to create initial segments in the image. Finally, a region-merging process is performed using the Bhattacharyya distance [23].

- Spectral and spatial information

  Recently, the increase of the spatial resolution has made interesting the combination of spectral and shape information to merge the segments. In [5], the authors propose a region-merging algorithm based on spectral and spatial information. The criterion used is called the Baatz & Schäpe criterion. Each segment R_i is described by spectral and spatial attributes. The spectral attribute is the vector of standard deviation values of the pixels contained in R_i denoted by σ_i. The spatial attributes are the area a_i, which is the number of pixels contained in R_i; the perimeter p_i; and its rectangular bounding box b_i, whose sides are aligned with the image axes. The cost of fusion between two adjacent segments R_i and R_j is denoted by $h_{i,j}$ and represents the increase in heterogeneity. Both segments R_i and R_j are merged if $h_{i,j} < s^2$, where s is a scale parameter. $R_{i,j}$ denotes the resulting segment from the fusion of R_i and R_j.

  The spectral increase of heterogeneity based on the standard deviation of R_i and R_j is noted by

$$h_{\text{spec}_{i,j}} = (a_i + a_j)\cdot\sigma_{i,j} - (a_i\cdot\sigma_i + a_j\cdot\sigma_j) \tag{1}$$

  where $\sigma_{i,j}$ is the standard deviation vector of the pixels contained in $R_{i,j}$.

  The shape component $h_{\text{shape}_{i,j}}$ is the spatial increase of heterogeneity based on the degree of smoothness and compactness. The degree of smoothness h_{smooth} is defined as the ratio between the perimeter p_i of the segment and the length $\text{len}(b_i)$ of its bounding box. The degree of compactness h_{compact} is defined as the ratio between the perimeter of the segment and the square root of its area. The expressions of the increase of both degrees when merging R_i and R_j are

$$h_{\text{smooth}_{i,j}} = \frac{a_{i,j}\cdot p_{i,j}}{\text{len}(b_{i,j})} - \left(\frac{a_i\cdot p_i}{\text{len}(b_i)} + \frac{a_j\cdot p_j}{\text{len}(b_j)}\right) \tag{2}$$

$$h_{\text{compact}_{i,j}} = \frac{a_{i,j}\cdot p_{i,j}}{\sqrt{a_{i,j}}} - \left(\frac{a_i\cdot p_i}{\sqrt{a_i}} + \frac{a_j\cdot p_j}{\sqrt{a_j}}\right). \tag{3}$$

  Then, the total spatial increase of heterogeneity is

$$h_{\text{shape}_{i,j}} = w_{\text{cpt}}\cdot h_{\text{compact}_{i,j}} + (1 - w_{\text{cpt}})\cdot h_{\text{smooth}_{i,j}} \tag{4}$$

  where w_{cpt} ranges from 0 to 1, indicating the importance of the degree of compactness relative to the degree of smoothness. Finally, the global increase of heterogeneity when merging these two segments is

$$h_{i,j} = w_{\text{spec}}\cdot h_{\text{spec}_{i,j}} + (1 - w_{\text{spec}})\cdot h_{\text{shape}_{i,j}} \tag{5}$$

  where w_{spec} ranges from 0 to 1, indicating the weight of the spectral component relative to the spatial component. To limit the undersegmentation, the user has to choose s, which influences the size of the resulting segments and the values of w_{spec} and w_{cpt}, to adjust the relative weight of the spectral and shape components and between the compactness and the smoothness degrees. The merging steps are repeated until there are no more possible merges. In [24], the authors propose a criterion based on spectral and spatial information called the full lambda schedule algorithm. The merging cost is expressed as follows:

$$\frac{\frac{a_i\cdot a_j}{a_i+a_j}\cdot\|M_i - M_j\|^2}{\text{len}(\partial(R_i,R_j))} < \lambda \tag{6}$$

  where R_i is the segment i of the image, a_i is the area of R_i, M_i is the average vector of spectral values of R_i, $\text{len}(\partial(R_i,R_j))$ is the length of the common boundary of the segments R_1 and R_j, and λ is a threshold.

  Larger values of λ produce larger segments with smaller common borders, and smaller values produce smaller segments with larger common borders.

  In [25], the authors propose a multistage segmentation process by first selecting the seeds using the gradient of the image. Next, a region-growing approach based on spectral and morphological information is applied using these seeds. Finally, a region-merging process based on spectral and morphological information is applied to refine the segments obtained from the region-growing procedure. We can also cite the work in [26], where the authors propose a segmentation algorithm combining spectral and morphological components to form a hierarchy of segments for each band of the image. A generic algorithm is then used to select the optimal segments, which correspond to actual objects in the scene. The hierarchical partition of the image is achieved by using a statistical clustering algorithm based on the Kullback–Leibler divergence.

2) Overview of Different Heuristics for Region Merging: For a given segment, there exist different ways to choose the adjacent segment to be merged with. Four possibilities are described in [5] as follows.

- Fitting (F): A segment R_1 is merged randomly with one of its adjacent segments R_2, for which the homogeneity criterion is fulfilled.
- Best Fitting (BF): For a segment R_1, we retain one of its adjacent segments R_2, for which the homogeneity criterion is fulfilled best. It represents the most similar adjacent segment.
- Local Mutual Best Fitting (LMBF): For a segment R_1, we determine its most similar adjacent segment R_2. For R_2, we determine its most similar adjacent segment R_3. R_1 and R_2 are merged if $R_3 = R_1$.

- Global Mutual Best Fitting (GMBF): At each iteration, we merge the pair of adjacent segments in the whole image, which fulfills the homogeneity criterion best.

Based on [5], we selected LMBF since it allows a symmetric growth of the segments, while minimizing the global heterogeneity within the final segments of the image. The authors in [27] claim that the use of this heuristic provides higher quality results. Another interesting aspect is the locality of the required neighborhood for each segment at each iteration. The size of this neighborhood can therefore be computed in advance. For any segment, it corresponds to the adjacent segments and the adjacent segments of those adjacent segments. Finally, this heuristic allows overcoming the issue of visiting order. If we consider the BF heuristic, we notice that a different visiting order may produce different resulting segments. For instance, let R_1, R_2, and R_3 be three segments. Let R_2 be the best segment of R_1, R_3 the best segment of R_2, and R_2 the best segment of R_3. If we visit the segments in this order $R_1 \rightarrow R_2 \rightarrow R_3$, then we have R_1 and R_2, which will merge, but R_3 will not merge since R_2 has already merged and does not exist anymore. If we visit the segments in this different order $R_2 \rightarrow R_3 \rightarrow R_1$, then we have R_2 and R_3, which will merge, but R_1 will not. As a consequence, the resulting segments are different. If we consider the LMBF heuristic and the previous visiting orders, we obtain the following for both the same resulting segments: the segment from the fusion of R_2 and R_3 and the segment R_1. In fact, the segments will always be the same for any visiting order since this heuristic is bijective. Indeed, the decision to merge is taken only if two segments are mutually the best, which avoids the fusion of one of these segments with any other region.

As we just saw in this section, a region-merging algorithm can be modeled by generic operations on a graph data structure. Each segment represents a node in the graph, and each edge represents a link between two adjacent segments. Each node is tagged with specific attributes according to the homogeneity criterion, and each edge contains three attributes: a pointer to a neighboring node, the merging cost, and the common boundary length with the neighboring node. Given the way to update the specific attributes and how to compute the merging costs, the iterative procedure of a region-merging algorithm consists of computing the merging costs between the adjacent nodes and merging the best pairs of nodes using the LMBF heuristic. As a consequence, a generic free-criterion region-merging library has been developed in the frame of this work[1] and will be soon integrated as an external module of the Orfeo Toolbox image processing software[2].

III. Proposed Solution

A. *Stability Margin for Segmentation Algorithms and Its Expression for Region-Merging Algorithms*

1) Overview of the Definition of Stability: The goal of this work is to ensure the equivalence of the result when applying a segmentation with and without tiling. As described in Section II, one experimental way to prove this equivalence is to use the Hoover metrics and check that $RC = 1$. The reference segmentation is the segmentation of the whole image at once, and the test segmentation is the tiled segmentation. There is equivalence of the results, when each segment obtained from the reference segmentation matches a segment from the test segmentation. In [4], the procedure to ensure that this property is fulfilled consists of stabilizing segmentation algorithms. The authors define two stability properties called the “inner” and “cover” properties. The inner stability property implies that each segment inside a tile matches a segment from the reference segmentation. The cover stability property implies that segments located on the tile edges are fully included in a segment from the reference segmentation. Fig. 2 illustrates the cover stability property. More formally, let I denote an image and $S : I \rightarrow S(I)$ represent a segmentation algorithm. $S(I)$ forms a partition of I into homogeneous disjoint segments $(R_1, \ldots, R_n)$, where R_i is one of these segments. $T \subset I$ denotes a tile, i.e., an image subset of I. A segment R' from the segmentation of T is represented by $R' \in S(T)$. For a segment $R \in S(I)$ and a tile $T \subset I$, we define $S_R(T)$ as follows:

$$S_R(T) = \{R' \in S(T) \setminus R' \subseteq R\}. \quad (7)$$

$S_R(T)$ is the set of segments $R' \in S(T)$ that are fully included in segment $R \in S(I)$. With these notations, the authors in [4] define a stable segmentation algorithm as follows:

Definition 1: Algorithm S is said to be stable if $\forall R \in S(I)$ and $\forall T \subset I$, the following properties hold:

$$R \subset T \Rightarrow \exists R' \in S(T) \setminus R' = R \quad (8)$$

$$R \cap T \neq \emptyset \Rightarrow R \cap T = \bigcup_{R' \in S_R(T)} R'. \quad (9)$$

Equation (8) represents the inner stability property, and (9) represents the cover stability property.

To make the mean-shift algorithm stable, the authors define an additional margin for each tile to ensure equivalence with the reference segmentation. Similarly, a margin will be defined for the generic region-merging algorithm to make it stable. In order to propose an expression for this margin, we first need to study the impact of image tiling on region-merging algorithms and determine the sources of the instability.

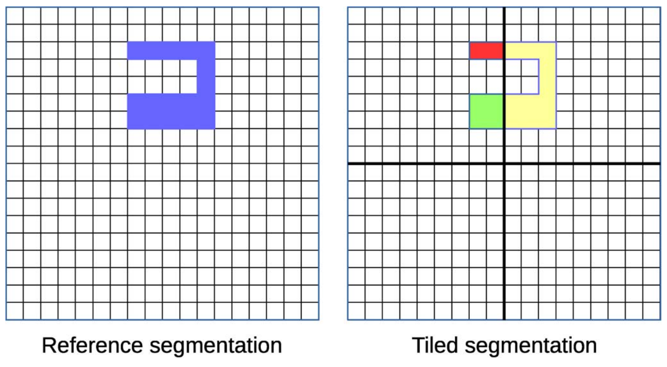


Fig. 2. Cover stability property. The segment from the reference segmentation is the union of segments from the tiled segmentation located on the tile edges.

[1]http://tully.ups-tlse.fr/lassallep/grm
[2]http://www.orfeo-toolbox.org/

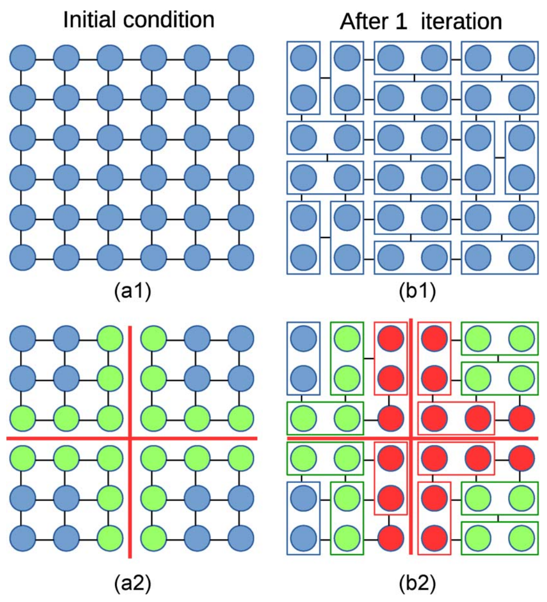


Fig. 3. Impact of image tiling on a region-merging segmentation result. (a2) Green nodes represent the nodes for which the set of edges is different due to the tile division. (b1) Region-merging iteration has been performed on the entire graph. (b2) Region-merging iteration has been performed on each subgraph. Segments different from the ones in (b1) are red. The green segments in (b2) are the segments for which the neighborhood is different.

2) Data-Driven Nature of Region-Merging Segmentation Algorithms and Impact of Image Tiling: As described in Section II, image tiling has an impact on the resulting segments. The region-merging procedure can be seen as a succession of operations on a graph. At the beginning of the segmentation procedure, each node represents a segment of one pixel and has four or eight edges depending on the choice of the neighbor connectivity. During the different stages of the algorithm, the graph is modified, as some pairs of segments are merged at each iteration. Operations on a node at a certain iteration require transforming and getting information from other nodes and edges. A decision to merge two segments leads to the fusion of two nodes in the graph. All the nodes have to be explored at each iteration to determine whether they have to be merged or not. This implies that a new iteration can be performed on the graph if all the nodes were processed at the previous iteration. Based on these characteristics, region-merging algorithms belong to the irregular data-driven algorithms category [28].

Applying image tiling on an initial graph modifies the initial set of edges for the nodes located on the borders of the tiles. After having performed one iteration of the region-merging procedure, these nodes might merge with different nodes from the ones expected and lead to different resulting segments. As a consequence, the neighborhood of other additional segments will be different. This impact might be propagated to other segments over the segmentation procedure. Fig. 3 illustrates the impact of image tiling on a region-merging segmentation result. The study of this impact of image tiling was first introduced in [7], where the green segments in Fig. 3 were qualified as contagious. Also, in [9], the authors explain that image tiling leads to nonoptimal fusions of segments due to the absence of knowledge of some segments, which belong to other tiles. Indeed, the red segments in Fig. 3(b2) result from nonoptimal fusions.

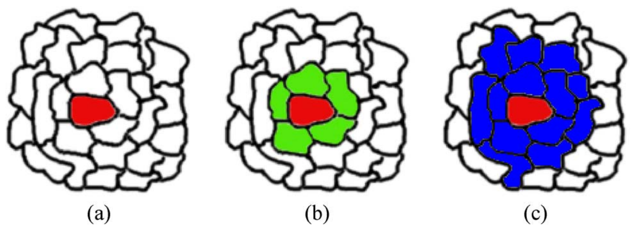


Fig. 4. Stability margin for one iteration of the region-merging procedure. (a) Represents the segment R. (b) Represents the list of adjacent segments of R. The best adjacent segment of R is one of the green segments. (c) Represents the union of the list of adjacent segments of the adjacent segments of R. This union $N(R)$ is necessary to fulfill the condition of mutuality.

We therefore define a stability margin for each tile to guarantee that each segment inside the tile will not be contagious and will not lead to a nonoptimal merge.

3) Expression of the Stability Margin for Region-Merging Algorithms: For a given segment R, its best adjacent segment will not be found farther from its list of adjacent segments. Since the LMBF heuristic is used, the list of all the adjacent segments of each adjacent segment of R is needed to fulfill the condition of mutuality. Considering the union of all these segments for each segment ensures the stability of the region-merging segmentation to perform one iteration. This union is composed of $N(R)$ and is illustrated in Fig. 4. At the beginning of the region-merging procedure, each segment is one pixel. The stability margin to perform the first iteration is therefore a crown of two pixels around each segment. Please note that this stability margin is the same whether we use a four- or eight-neighbor connectivity. To ensure the stability for the first iteration of the region-merging procedure, it is sufficient to a add a margin of two pixels to each tile. The size of the margin depends on the number of iterations and can be determined by recurrence. Let M_n be the size of the margin to perform the first n iterations. The objective is to determine the value of M_{n+1}. Let R be a segment after having performed the n first stable iterations. To perform an additional stable iteration, $N(R)$ must be considered as explained previously. To ensure the stability for this additional iteration, M_n must have been considered for each segment contained in $N(R)$. The upper bound for the number of pixels contained in a segment after n iterations is equal to 2^n. Indeed, this upper bound is reached if the segment merges at each iteration with a segment containing the maximum number of pixels. According to Fig. 5, M_{n+1} can be expressed in function of M_n as follows:

$$M_{n+1} = 2^{n+1} + M_n, \text{ with } M_0 = 0 \tag{10}$$

which can be written as

$$M_{n+1} = 2^{n+2} - 2. \tag{11}$$

For example, the margin of stability to perform the two first iterations is a crown of $M_2 = 6$ pixels around each segment. To ensure the stability for the two first iterations of the region-merging procedure, it is sufficient to a add a margin of 6 pixels to each tile.

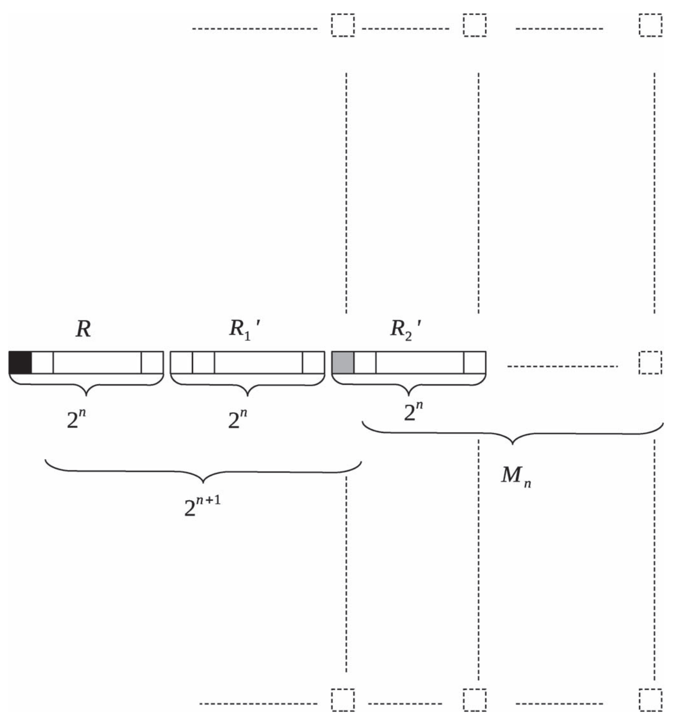


Fig. 5. Determination of the size of the margin by recurrence. R is a segment obtained after n iterations of the region-merging procedure. R'_1 is its adjacent segment, and R'_2 is an adjacent segment of R'_1. R'_1 and R'_2 are included in $N(R)$ and are needed to ensure the stability for an additional iteration for R. R and all the segments included in $N(R)$ contain at maximum 2^n pixels after n iterations. A margin of size M_n has to be considered for all the segments in $N(R)$ to ensure the stability after n iterations. M_{n+1} can be expressed as a function of M_n as follows: $M_{n+1} = 2^{n+1} + M_n$.

4) Definition of a Stable Segment and Removal of the Unstable Segments: The initial segments from the margin are processed by the region-merging algorithm to ensure the stability of the segments inside the tile. However, as explained in Section III-A2, these segments may be contagious or may result from nonoptimal fusions. These segments must not be taken into consideration and must be removed from the graph. A segment is said to be stable if it contains at least one pixel inside the tile, since a margin is considered for this pixel.

We consider an image I and a tile $T \subset I$. T is a rectangular area defined by four bounds: the upper and lower rows denoted by $r_{\max}$ and $r_{\min}$ and the upper and lower columns $c_{\max}$ and $c_{\min}$. A margin is added to T to perform n iterations. After the segmentation of T over n iterations, we obtain a graph containing both stable and unstable segments. The bounding box of the segments is first analyzed. A bounding box represents the smallest rectangle enclosing the segment. It is characterized by the upper left coordinates u_x, u_y, its width bb_w, and its height bb_h. If the following condition is true:

$$\begin{cases} u_x \geq c_{\min} & \text{and} \\ u_y \geq r_{\min} & \text{and} \\ u_x + bb_w \leq c_{\max} & \text{and} \\ u_y + bb_h \leq r_{\max} \end{cases} \tag{12}$$

then the segment is fully included inside T and is stable. At the opposite, if the following condition is true

$$\begin{cases} u_x > c_{\max} & \text{or} \\ u_y > r_{\max} & \text{or} \\ u_x + bb_w < c_{\min} & \text{or} \\ u_y + bb_h < r_{\min} \end{cases} \tag{13}$$

then the segment is unstable and removed from the graph. However, for a bounding box that overlaps the borders of T, no decision can be taken. In this case, the list of the border pixels of the segments is explored. The segment is stable if one of its border pixels is inside the tile.

B. Tile-Based Framework for Region-Merging Segmentation of Large Images

1) High-Level Overview: To simplify our discussion, we initially walk through the main steps of the entire approach. In the remainder of this paper, we consider a standard computer with two types of memory: the Quick Limited Storage denoted as QLS, which is typically the random access memory (RAM), and the Slow Unlimited Storage denoted as SUS, which can represent the hard disk drive. The objective of this section is to exhibit how the framework works when the QLS limits the region-merging algorithm to operate on a graph of maximum N segments. The flow diagram is described in Fig. 6. The new algorithm breaks up the process of region merging into successive partial segmentations of the graphs of segments of the tiles.

The first step consists of determining the size of the margin for each tile, knowing the capacity of the QLS. Different strategies are possible for choosing the size of the margin. Increasing the margin implies a higher reduction of the number of segments since more iterations of the region-merging procedure can be performed. However, this strategy implies a higher number of tiles since their sizes are smaller. More I/O operations are therefore necessary to load and store the tiles on the SUS. Increasing the size of the tiles implies a smaller size of the margin and, hence, a lower reduction of the number of segments. This strategy implies more iterative partial segmentations of the graphs of the tile. The problem of finding the best tradeoff is not tackled in this paper but should be addressed in the future.

The second step consists of the initial partial segmentation of the tiles over n_1 iterations. As a result of these segmentations, graphs of segments are obtained and contain both stable and unstable segments. Unstable segments must be removed from these graphs, as explained in Section III-A4. These graphs are then written to files and stored on the SUS. Section III-B2 will explain in detail how these graphs are stored on the SUS.

The third step is a loop where successive partial segmentations of the graphs are performed over n_2 iterations. Currently, $n_2 = 1$, but further improvements will be achieved to make possible a higher number of iterations for each partial segmentation. The loop stops when all the segments can be stored on the QLS or when there are no remaining segments to be merged for each graph. The body of the loop consists of

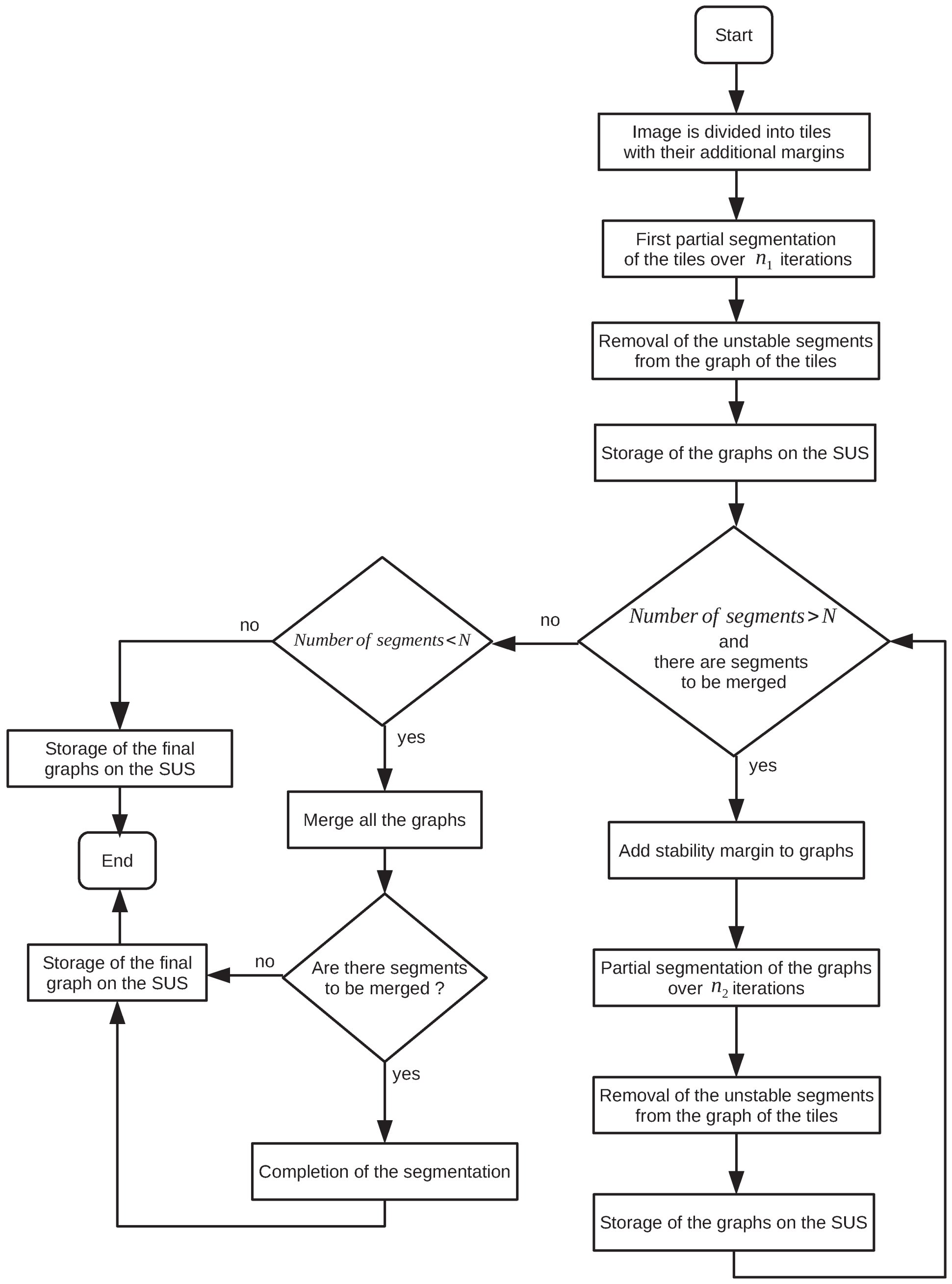

Start
Image is divided into tiles with their additional margins
First partial segmentation of the tiles over n_1 iterations
Removal of the unstable segments from the graph of the tiles
Storage of the graphs on the SUS
Number of segments > N and there are segments to be merged
no
yes
Add stability margin to graphs
Partial segmentation of the graphs over n_2 iterations
Removal of the unstable segments from the graph of the tiles
Storage of the graphs on the SUS
Number of segments < N
no
yes
Storage of the final graphs on the SUS
End
Merge all the graphs
Are there segments to be merged ?
no
yes
Storage of the final graph on the SUS
Completion of the segmentation

Fig. 6. Flow diagram of the tile-based framework for region-merging segmentation algorithms.

adding a stability margin to each graph to ensure the stability for n_2 iterations. The graphs are segmented over n_2 iterations and, as described before, the unstable segments are removed, and then, the graphs are stored on the SUS.

This third loop stops if the total number of segments is smaller than N or if there are no remaining pairs of segments to be merged. If the number of segments is still greater than N but there are no remaining pairs of segments to be merged, the procedure is complete and the graphs of the tiles are stored on the SUS. If all the segments of the graphs can be stored on the QLS, the last step consists of merging all the graphs to form one graph of the segments of the image. If there are no remaining pairs of segments to be merged, then the final graph is stored on the SUS; otherwise, the segmentation is achieved on this graph and stored on the SUS.

2) Storing and Loading the Graph: After the partial segmentation of a tile, a graph of segments is obtained. An adjacency list is used to represent the graph of segments, where each node represents a segment and each edge represents a link between two adjacent segments. In addition to specific attributes, a node contains a list of outedges, where each outedge targets a neighboring node. To store the graph on the SUS, a key is used to uniquely identify each node. In a first file, each node is written as follows:

$$(k_i, a_1, a_2, \ldots, a_k)$$

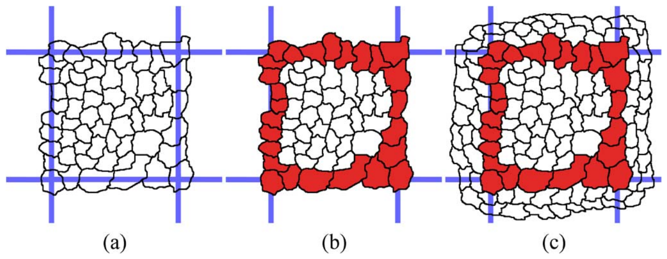


Fig. 7. Graph of stable segments and the construction of its stability margin for one iteration. (a) Represents the graph of stable segments loaded from the SUS. (b) Highlights the border segments located on the tile borders. (c) Represents the stability margin for one iteration for all the border segments, which ensures that we are considering also this stability margin for all the segments within the stable area. The stability margin corresponds to the second neighboring layer $N_2(R)$ for each stable segment R of the graph.

where k_i is the key of the ith node in the graph, and a_j with $j \in \mathbb{N}^*$ is one of its specific attributes. In a second file, for each node, its list of adjacent nodes and the specific attributes of each outedge is written as follows:

$$(k_i, [k_{i1}, e_1, \ldots, e_k], \ldots, [k_{ie}, e_1, \ldots, e_k])$$

where k_{ij} is the key of the adjacent node of the ith in the graph, and e_j is one of the specific attributes of an outedge.

Loading a graph of segments for a new partial segmentation consists of two steps:

1) building the adjacency list of the nodes from the first binary file;
2) building the list of edges of each node from the second binary file.

During the construction of the nodes, a hashtable is used to associate, for each key, the pointer to the corresponding node. During the creation of the edges, the pointer to the node can be retrieved knowing its key using the hashtable. This operation is achieved in constant time.

3) Addition of a Stability Margin to a Graph: To perform a new partial segmentation on a graph over n_2 iterations, we have to consider a stability margin of value M_{n_2} for each stable segment. Since $n_2 = 1$, it results in adding $N(R)$ for each segment R of the graph, as explained in Section III-A3. Some of these segments are already contained in the graph. However, for the segments located on the borders of the graph, some segments contained in $N(R)$ are missing, as shown in Fig. 7. Indeed, they are contained in the adjacent graphs of the graph. When storing a graph on the SUS, another file is used to store a subgraph containing only the segments located on the borders and their sets of segment $N(R)$. This subgraph will be read to create the stability margins of the adjacent graphs. Adding a stability margin to a graph results in merging the graph with subgraphs. The way to merge graphs is described in Section III-B4.

4) Merging the Graphs: Merging two or more graphs can happen when a stability margin is added to a graph, as described in the previous section, or when the graphs of the tiles are merged to form the graph of segments of the input image.

Fig. 8(a) represents a pair of graphs to be merged. The segments that overlap the common borders between the adjacent tiles are duplicated [see Fig. 8(b)]. Let N_d be a segment and L_{Nd} be the list of its duplicated segments. The merging operation consists of updating the list of the edges of N_d by exploring the list of the edges of the nodes contained in L_{Nd} [see Fig. 9(a)]. Once the edges of N_d are updated, the nodes contained in L_{Nd} can be removed [see Fig. 9(b)].

Another required operation is to detect the segments that contain pixels exactly on one side of the borders without overlapping. Their neighborhoods have to be updated by detecting the adjacent segment on the other side of the borders. An edge is then added between their corresponding nodes [see Fig. 8(c)].

IV. EXPERIMENTS

A. Experimental Verification of the Stability Margin

The first experiment aims at verifying the correctness of the stability margin. We want to show that considering the stability margin for a tile ensures that the region-merging segmentation algorithm meets the condition of the stability properties described in Section III-A1. To do so, we denote I as a 500 × 500 pixel subset from a scene. We use two different scenes provided by two different optical satellites: Ikonos and Pléiades. We use for each image the three homogeneity criteria described in Section II-B1.

For each criterion, we apply the following procedure.

1) The segmentation of the whole image I is performed at once. We call G_T the segmentation result. We extract from G_T a tile of size 250 × 250 pixels. We call it E_T.
2) We extract a tile from the image covering the same zone as E_T. We consider for this tile successive margins with ascending values. The range of the values is between 0 and 250 pixels (whole image) with a step of 50. The successive segmentation results of the tiles $(S_0, S_{50}, \ldots, S_{250})$ are compared to E_T using the Hoover metrics [6], which was described previously in Section I.

For both images, Fig. 10 shows the evolution of RC according to the margin value for a given homogeneity criterion with specific segmentation parameters. One can see that, for both cases, RC converges to 1 with increasing values of the margin for any criterion. RC is equal to 1 when the margin is equal to 250 pixels, which is expected, since it corresponds to the whole image. The speed of convergence of RC depends on the selected criterion and also on the image. According to Fig. 10, we can observe that, for the FLS criterion and the Baatz & Schäpe criterion, the correct detection score is equal to 1, before the tile with its additional margin is equal to the whole image. For example, for the FLS criterion and the Ikonos scene, considering the tile with a margin of 100 pixels is enough to limit the “contagious property” of being propagated to the segments containing pixels within the tile. For the Baatz & Schäpe criterion and the Ikonos scene, it is necessary to consider a margin of 200 pixels to ensure the stability. For the Euclidean distance criterion and the Ikonos scene, the stability of the segments within the tile is never ensured until the tile with its margin is equal to the whole image. When we compare the evolutions of RC for both images, we can see that they have the same relative order of convergence speeds between the

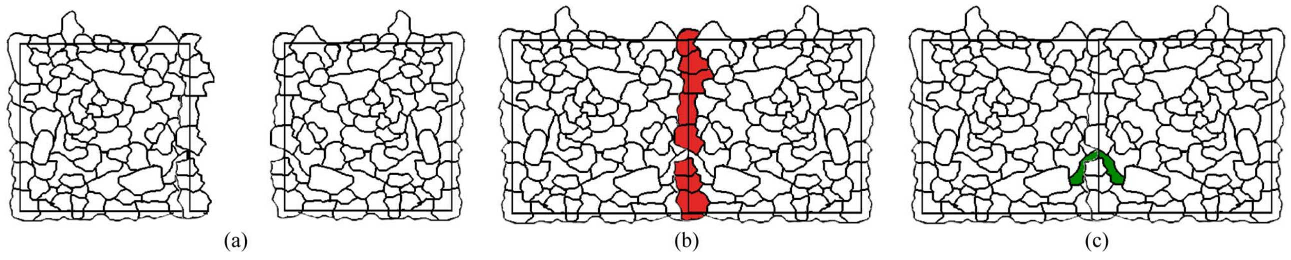


Fig. 8. Aggregation of two graphs of segments. (a) Graphs to be aggregated. (b) Processing of duplicated segments. (c) Update of missing edges.

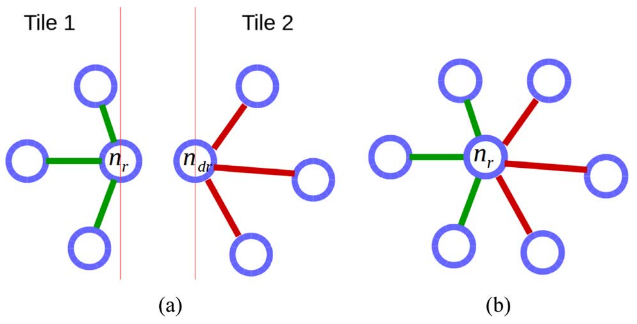


Fig. 9. Graph operation for duplicated segments.

criteria. However, for a given criterion and for different images, the values of RC are not identical according to the values of the margin. For instance, for the Baatz and Schäpe criterion, a margin of 100 pixels is enough for the Pléiades image [see green line in Fig. 10(b)], but it would not be enough for the Ikonos image [see green line Fig. 10(a)]. Therefore, it appears difficult to try to anticipate the correct value of the margin for a new unknown image. That is the reason why we express a generic stability margin for any particular criterion and scene, which ensures the stability of the segments within the tiles.

The following experiment will check that our tile-based strategy for segment-merging algorithms described in Section II-B1 ensures identical results to those obtained when the complete image is segmented. To do so, I is divided into tiles of size 250×250 pixels. For each tile, we consider an additional stability margin M_n, where n is the number of iterations that we anticipate. First, a segmentation of n iterations is performed on each tile with $n \in [1, 6]$. Next, the intermediate graphs of segments are merged together, and the segmentation is achieved. We call S_{M_n} the segmentation result. S_{M_n} is then compared to G_T using the Hoover metrics. For each criterion and for each scene, a correct detection score equal to 1 is obtained.

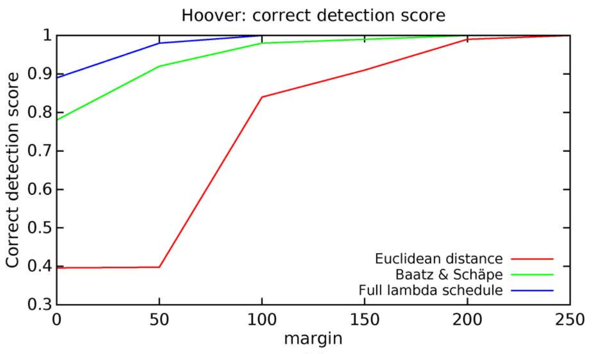
B. Illustration of the Main Steps of the Tile-Based Region-Merging Framework

We consider a 1000×1000 QuickBird scene and the Baatz and Schäpe criterion [5] for our region-merging algorithm with a spectral and shape weight of 0.7 and 0.3, respectively, and a scale threshold of 150. We simulate a computer with 12 MB of RAM. The first step consists of determining the size of the tiles and the size of the margin. To do so, we have to know the sizes of an initial node and edge in the graph. Each node represents a region and contains several attributes to compute the merging costs, as described in Section II-B1. Likewise, each edge contains three attributes: the merging cost, the boundary length, and a pointer to the neighboring nodes. With the features of the simulated computer, the size of a node is 216 bytes, and the size of an edge is 24 bytes. We use the four-neighbor connectivity, which implies that, at the beginning, each node has four edges. If N is the number of pixels in a tile, then the required memory to store the corresponding graph is equal to $(216 + 4 \times 24) \times N$ bytes. A size of 125×125 pixels for the tile and a margin value of 30 pixels results in 38 025 pixels. The memory required to store the initial graph of each tile with its margin is 11 MB, which can fit in our computer. The image is therefore divided into 64 tiles with their stability margins. Fig. 11 illustrates the division strategy.

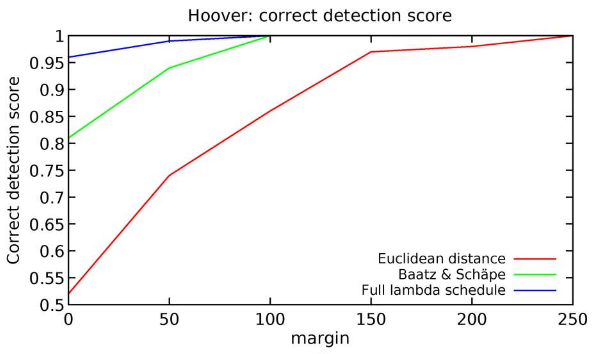
As described in Fig. 6, the next step consists of performing the first partial segmentation of the tiles over 4 iterations $(M_4 = 30)$. At the end of this step, the unstable segments are removed from each graph, and the graphs of the tiles are stored on the SUS. The accumulated memory to store all the intermediate graphs is equal to 90 MB, which is still greater than the available QLS, and there are remaining pairs of segments to be merged in each graph. The next step consists of entering in the main loop of the framework, which consists of performing successive iterations on the graphs of the tiles, while the memory to store all the segments of the image is greater than 12 MB, and there are remaining pairs of segments to be merged.

Twelve additional partial segmentations have been performed on the graphs of the tiles to be able to store all the segments of the image in the QLS. Before each partial segmentation, a margin for one iteration has been added to each graph, as described in Section III-B3. After a partial segmentation, the unstable segments are removed, and the graph is stored on the SUS, as described in Fig. 6, in the body of the main loop. Fig. 12 shows the evolution of a graph of segments of a tile over the successive partial segmentations. Only four states of the stable graph of this tile are illustrated and correspond to the first, fourth, seventh, and thirteenth partial segmentation, respectively.

The final step consists of merging all the intermediate graphs together to form the global graph of the image. Since there are remaining segments to be merged, we perform and achieve the segmentation on the global graph. Fig. 13(a) shows the labeled segmented image. We notice the absence of the artifacts at the tile boundaries. Furthermore, we compare this segmentation result with the one obtained from the segmentation of the whole image without tiling [see Fig. 13(b)] using the Hoover metrics [6]. A correct detection score of 1 has been obtained, which means that both segmentations are identical.



(a)



(b)

Fig. 10. Correct detection scores: Comparison of the test segmentations $(S_0, S_{50}, \ldots, S_{250})$ to E_T using three different homogeneity criteria when performing the step 2 of the procedure. (a) and (b) Evolution of the detection scores using two different images (Ikonos and Pléiades extracts). (a) Ikonos. (b) Pléiades.

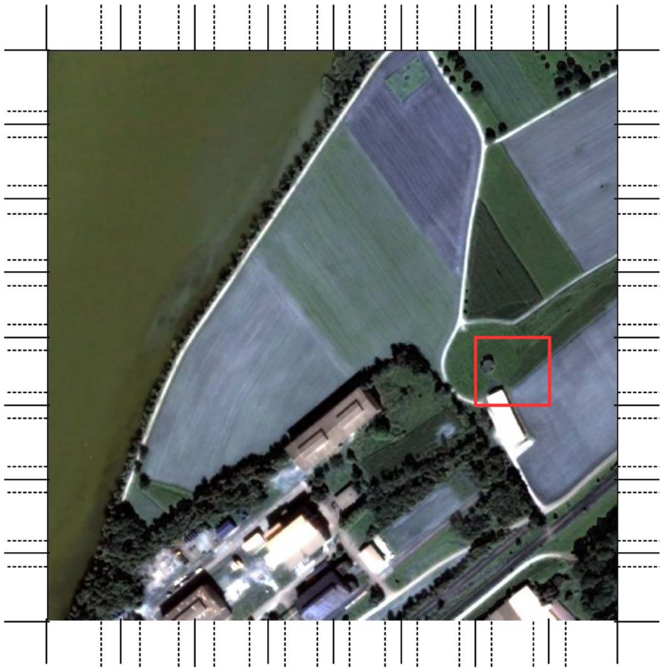


Fig. 11. Division of the image scene of size 1000 × 1000 pixels into 64 tiles of size 125 × 125 pixels: the tiles are delineated by continuous black borders and the stability margins are delineated by the dashed black borders. The red rectangle represents one of the tiles of the image and will be used in Fig. 12 to show the evolution of its corresponding graph of segments over the procedure of the framework described in Fig. 6.

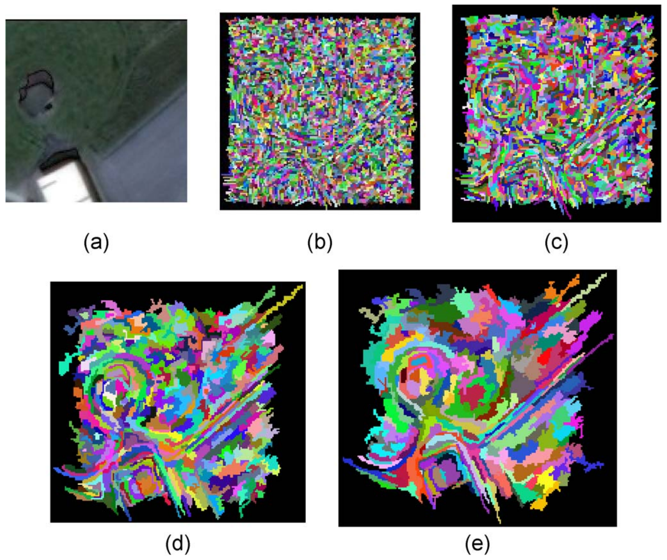


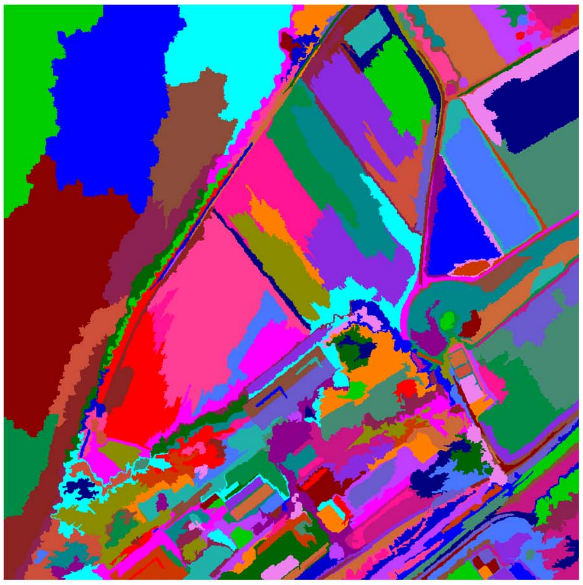
Fig. 12. Evolution of a graph of a tile over the partial segmentations. (a) Tile of the input image represented by a red rectangle in Fig. 11. (b) Graph of stable segments of the tile after the first partial segmentation over 4 iterations. (c) Graph of stable segments of the tile after the fourth partial segmentation. (d) After the seventh partial segmentation. (e) After the last partial segmentation. The growth of the segments can be observed over the iterations, and segments located at the tile edges have the freedom to grow in the same way as they would do if the input image was segmented without image tiling, due to the stability margin.

C. Segmentation of a Full Pléiades Scene Using the Tile-Based Region-Merging Framework

Here, we scale our tile-based region-merging framework to a full Pléiades pan-sharpened scene of Melbourne city in Australia. This image contains 32 768 × 16 384 pixels. The segmentation was performed using the Baatz and Schäpe homogeneity criterion [5]. The segmentation was performed sequentially on an Intel Xeon with 12 GB of RAM. For the first partial segmentation, the image was divided into 512 tiles of 1024 × 1024 pixels with an additional stability margin of 1022 pixels corresponding to 9 iterations. The first partial segmentation was performed on the 512 tiles over 9 iterations. The accumulated memory of all the intermediate graphs after the first partial segmentations was equal to 31 GB, which was larger than the RAM available. Twelve more iterations were performed to be able to store the global graph of segments in QLS. The graphs of segments were then merged, and the segmentation was achieved. The whole sequential procedure took approximately 12 h and 30 min as follows: 11 h for the successive partial segmentations and 1 h and 30 min to merge the intermediate graphs of segments and complete the segmentation. Fig. 14 shows the input image and one enlarged view of the output image. This enlarged view shows the contour of the segments.

V. Conclusion

This paper has presented a solution to ensure equivalent results for the segmentation of satellite images of arbitrary size with tiling. It was experimentally shown that the region-merging algorithms do not cope well when using image tiling. The impact of image tiling has been studied for region-merging algorithms, and the critical steps have been identified. The concept of stability margin has been defined and expressed quantitatively as a function of the number of iterations of


(a)

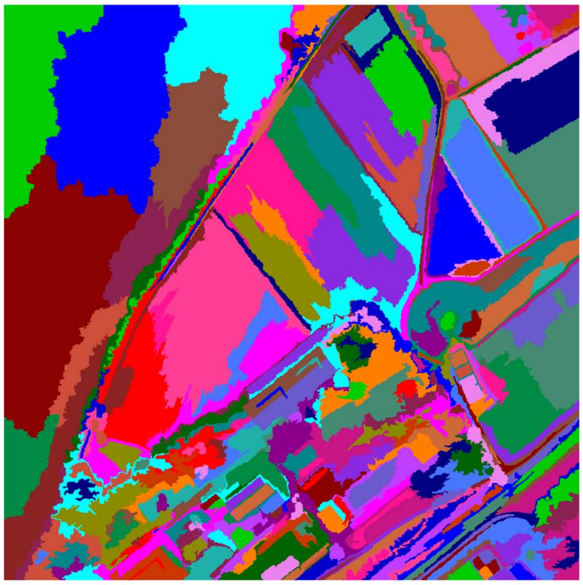
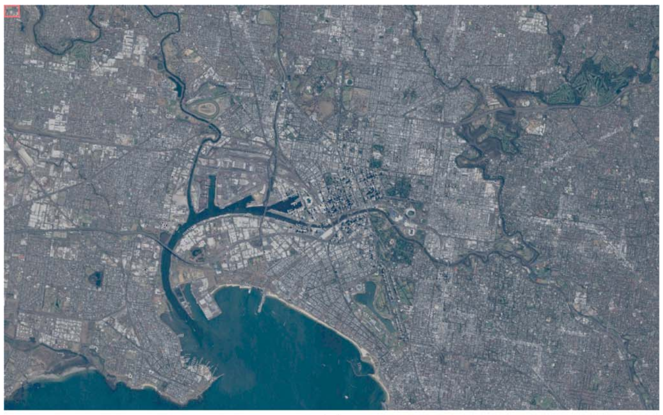

(b)

Fig. 13. (a) Labeled segmented image after applying our tile-based region-merging framework. No artifacts can be observed at the tile boundaries. (b) Labeled segmented image obtained without image tiling but with the same user-defined parameters for the Baatz & Schäpe criterion. Both results (a) and (b) have been compared with the Hoover metrics: $RC = 1$, $RA = 0$, $RF = 0$, and $RM = 0$. The equivalence of the result is proven. (For both images, the same seed has been used to randomly color the segments in order to have a perfect match.)


(a)

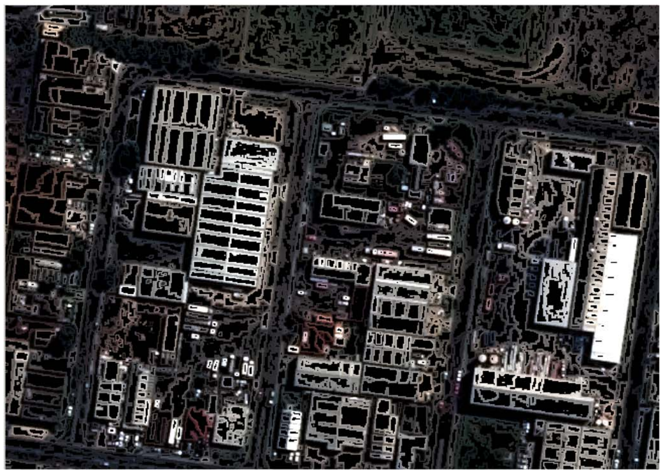

(b)

Fig. 14. (a) Full VHR Pléiades scene (32 768 × 16 384 pixels). The image was divided into 32 × 16 tiles of 1024 × 1024 pixels for our tiled-based strategy. (b) Partial views of the segmentation result. The segments are represented only by their edges.

the region-merging procedure. Using the stability margin in a tiling approach has allowed ensuring the equivalence of the results between the segmentation with and without tiling. The practicality of the tile-based framework for region-merging algorithms has been illustrated by the segmentation of a full entire Pléiades scene of billions of pixels in a computer with limited storage memory.

However, additional work to improve the efficiency of this framework, by minimizing the number of partial iterations and the number of I/O operations, should be carried on. The first improvement would consist of finding an optimal solution to determine the size of the tiles and the stability margin. The second improvement would consist of allowing more iterations for each partial segmentation of the graphs. Additional study may also include the portage of this framework to a parallel and distributed environment to reduce the processing time. Finally, it would be interesting to unify this solution, by extending it to other families of segmentation algorithms.

Acknowledgment

P. Lassalle would like to thank P. N. Happ, G. A. O. P. Costa, and R. Q. Feitosa for their introduction to the region-merging algorithms developed in the open-source software InterImage. All Pléiades images are a copyright of CNES (2012), Distribution Airbus DS/Spot Image.

References

[1] T. Blaschke, "Object based image analysis for remote sensing," *ISPRS J. Photogramm. Remote Sens.*, vol. 65, no. 1, pp. 2–16, Jan. 2010.

[2] J. Inglada and J. Michel, "Qualitative spatial reasoning for high-resolution remote sensing image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 2, pp. 599–612, Feb. 2009.

[3] M. Vanegas, I. Bloch, and J. Inglada, "Alignment and parallelism for the description of high-resolution remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 6, pp. 3542–3557, Jun. 2013.

[4] J. Michel, D. Youssefi, and M. Grizonnet, "Stable mean-shift algorithm and its application to the segmentation of arbitrarily large remote sensing

images," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 2, pp. 952–964, Feb. 2015.

[5] M. Baatz and A. Schäpe, "Multiresolution segmentation: An optimization approach for high quality multi-scale image segmentation," in *Angewandte Geographische Informationsverarbeitung XII*. Heidelberg, Germany: Wichmann, 2000, pp. 12–23.

[6] A. Hoover *et al.*, "An experimental comparison of range image segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 7, pp. 673–689, Jul. 1996.

[7] S. Lee, "An unsupervised hierarchical clustering image segmentation and an adaptive image reconstruction system for remote sensing," Ph.D. dissertation, Univ. Texas Austin, Austin, TX, USA, 1990.

[8] S. Lee, K.-H. Lee, and C. Kim, "Efficient multi-stage system for unsupervised classification and its application of KOMPSAT-I imagery," in *Proc. IEEE IGARSS*, 2000, vol. 5, pp. 2173–2175.

[9] J. C. Tilton, Y. Tarabalka, P. M. Montesano, and E. Gofman, "Best merge region-growing segmentation with integrated nonadjacent region object aggregation," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4454–4467, Nov. 2012.

[10] T. Körting, E. Castejon, and L. Fonseca, "The divide and segment method for parallel image segmentation," in *Advanced Concepts for Intelligent Vision Systems*, ser. Lecture Notes in Computer Science, J. Blanc-Talon, A. Kasinski, W. Philips, D. Popescu, and P. Scheunders, Eds. Cham, Switzerland: Springer-Verlag, 2013, vol. 8192, pp. 504–515.

[11] J. C. Tilton, G. Marchisio, K. Koperski, and M. Datcu, "Image information mining utilizing hierarchical segmentation," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2002, pp. 1029–1031.

[12] J. Michel *et al.*, "Open tools and methods for large scale segmentation of very high resolution satellite images," in *Proc. OGRS*, 2012, pp. 179–184.

[13] J. Schiewe, "Segmentation of high-resolution remotely sensed data-concepts, applications and problems," in *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. 34, no. 4, pp. 380–385, 2002.

[14] G. Meinel and M. Neubert, "A comparison of segmentation programs for high resolution remote sensing data," *Int. Arch. Photogramm. Remote Sens.*, vol. 35, Pt. B, pp. 1097–1105, 2004.

[15] C. R. Brice and C. L. Fennema, "Scene analysis using regions," *Artif. Intell.*, vol. 1, no. 3, pp. 205–226, 1970.

[16] R. Adams and L. Bischof, "Seeded region growing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 6, pp. 641–647, Jun. 1994.

[17] F. Calderero and F. Marques, "Region merging techniques using information theory statistical measures," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1567–1586, Jun. 2010.

[18] L. S. Bins, L. M. G. Fonseca, G. J. Erthal, and F. M. Ii, "Satellite imagery segmentation: A region growing approach," in *Proc. Simpósio Brasileiro Sensoriamento Remoto*, 1996, vol. 8, pp. 677–680.

[19] G. Espindola, G. Câmara, I. Reis, L. Bins, and A. Monteiro, "Parameter selection for region-growing image segmentation algorithms using spatial autocorrelation," *Int. J. Remote Sens.*, vol. 27, no. 14, pp. 3035–3040, Jul. 2006.

[20] G. Costa *et al.*, "Knowledge-based interpretation of remote sensing data with the Interimage system: Major characteristics and recent developments," in *Proc. ISPRS*, 2010, vol. 38, p. 4.

[21] C. Woodcock and V. Harward, "Nested-hierarchical scene models and image segmentation," *Int. J. Remote Sens.*, vol. 13, no. 16, pp. 3167–3187, Nov. 1992.

[22] S. Lee and C. Lee, "Unsupervised segmentation for hyperspectral images using mean shift segmentation," in *Proc. SPIE Opt. Eng. Appl.*, 2010, Art. ID. 781011.

[23] F. Calderero and F. Marques, "General region merging approaches based on information theory statistical measures," in *Proc. 15th IEEE ICIP*, 2008, pp. 3016–3019.

[24] D. J. Crisp, P. Perry, and N. J. Redding, "Fast segmentation of large images," in *Proc. 26th Australasian Comput. Sci. Conf.*, 2003, vol. 16, pp. 87–93.

[25] K. Segl, S. Roessner, U. Heiden, and H. Kaufmann, "Fusion of spectral and shape features for identification of urban surface cover types using reflective and thermal hyperspectral data," *ISPRS J. Photogramm. Remote Sens.*, vol. 58, no. 1/2, pp. 99–112, Jun. 2003.

[26] H. G. Akcay and S. Aksoy, "Automatic detection of geospatial objects using multiple hierarchical segmentations," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 7, pp. 2097–2111, Jul. 2008.

[27] P. N. Happ, R. Q. Feitosa, C. Bentes, and R. Farias, "A region growing segmentation algorithm for GPUs," *Boletim Ciências Geodésicas*, vol. 19, no. 2, pp. 208–226, 2013.

[28] K. Pingali *et al.*, "The tao of parallelism in algorithms," in *Proc. PLDI*, 2011, pp. 12–25.

Pierre Lassalle received the degree in informatics, automation, and embedded systems engineering from the École Nationale Supérieure des Techniques Avancées, Brest, France, and the degree in computational science and intelligent systems from the University of the Basque Country, San Sebastian, Spain, in 2012. He is currently working toward the Ph.D. degree with the Centre National d'Études Spatiales (French Space Agency) Toulouse, France, where he is performing research on the scalability of image processing algorithms for large remote sensing images.

Jordi Inglada received the degree in telecommunications engineering from both the Universitat Politècnica de Catalunya, Barcelona, Spain, and the École Nationale Supérieure des Télécommunications de Bretagne, Brest, France, in 1997 and the Ph.D. degree in signal processing and telecommunications from the Université de Rennes 1, Rennes, France, in 2000.

He is currently with the Centre National d'Études Spatiales (French Space Agency), Toulouse, France, working in the field of remote sensing image processing at the Centre d'Etudes Spatiales de la Biosphère (CESBIO) Laboratory. He is in charge of the development of image processing algorithms for the operational exploitation of Earth observation images, mainly in the field of multitemporal image analysis for land use and cover change.

Julien Michel (M'14) received the degree in telecommunications engineering from the École Nationale Supérieure des Télécommunications de Bretagne, Brest, France, in 2006.

From 2006 to 2010, he was with Communications et Systèmes, Toulouse, France, working on studies and developments in the field of remote sensing image processing. He is currently with the Centre National d'Études Spatiales (French Space Agency), Toulouse, France, where he is in charge of research and development on image processing algorithms for remote sensing images.

Manuel Grizonnet received the degree in mathematical modeling, vision, graphics, and simulation engineering from the École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble, Grenoble, France, in 2007.

From 2007 to 2009, he was with BRGM (French geological survey), Niamey, Niger, where he was a Systems and Geological Information System (GIS) Engineer in the framework of the SYSMIN project, which aimed to give the Niger ways to promote its mining potential by establishing a GIS. He is currently with the Centre National d'Études Spatiales (French Space Agency), Toulouse, France, where he is developing image processing algorithms and software for the exploitation of Earth observation images.

Julien Malik received the master's degree with a specialization in signal and image processing from the École Supérieure d'Electricité, Gif-sur-Yvette, France, in 2004.

After different positions where he developed image and signal processing algorithms in various contexts, since 2010, he has been with CS Systèmes d'Information, Toulouse, France, where he is working on R&D studies and software development in the field of remote sensing image processing. He is involved with the development of a number of open-source toolboxes for remote sensing imagery analysis.