

# Hierarchical Finite-State Modeling for Texture Segmentation with Application to Forest Classification

Giuseppe Scarpa, Michal Haindl, Josiane Zerubia

▶ **To cite this version:**

Giuseppe Scarpa, Michal Haindl, Josiane Zerubia. Hierarchical Finite-State Modeling for Texture Segmentation with Application to Forest Classification. [Research Report] RR-6066, INRIA. 2006, pp.47. inria-00118420v2

**HAL Id: inria-00118420**

**https://inria.hal.science/inria-00118420v2**

Submitted on 18 Dec 2006

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Hierarchical Finite-State Modeling for Texture Segmentation with Application to Forest Classification*

Giuseppe Scarpa — Michal Haindl — Josiane Zerubia

**N° 6066**

December 2006

Thème COG




ISSN 0249-6399 ISRN INRIA/RR--6066--FR+ENG



# Hierarchical Finite-State Modeling for Texture Segmentation with Application to Forest Classification

Giuseppe Scarpa*, Michal Haindl† , Josiane Zerubia

Thème COG — Systèmes cognitifs
Projets Ariana

Rapport de recherche n° 6066 — December 2006 — 47 pages

**Abstract:** In this research report we present a new model for texture representation which is particularly well suited for image analysis and segmentation. Any image is first discretized and then a hierarchical finite-state region-based model is automatically coupled with the data by means of a sequential optimization scheme, namely the *Texture Fragmentation and Reconstruction* (TFR) algorithm. The TFR algorithm allows to model both intra- and inter-texture interactions, and eventually addresses the segmentation task in a completely unsupervised manner. Moreover, it provides a hierarchical output, as the user may decide the scale at which the segmentation has to be given. Tests were carried out on both natural texture mosaics provided by the *Prague Texture Segmentation Datagenerator Benchmark* and remote-sensing data of forest areas provided by the French National Forest Inventory (IFN).

**Key-words:** Texture segmentation, classification, co-occurrence matrix, structural models, Markov chain, texture synthesis, forest classification.

This work was carried out during the tenure of an ERCIM fellowship (Scarpa's postdoc).

* ERCIM Fellow hosted by UTIA/CRCIM Prague (CZ) and INRIA Sophia Antipolis (F). Now he is an Assistant Professor of the University "Federico II" of Naples (I). Email: giscarpa@unina.it

† Pattern Recognition Department, Institute of Information Theory and Automation of the Czech Academy of Sciences, Prague (CZ). Email: haindl@utia.cas.cz

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

# Modèle hiérarchique à états finis pour la segmentation de texture. Application à la classification de forêts

**Résumé :** Dans ce rapport de recherche, nous proposons un nouveau modèle pour la représentation des textures qui est particulièrement bien adapté à l'analyse et à la segmentation des images. Chaque image est d'abord discrétisée. Ensuite, cette représentation discrète est automatiquement associée à un modèle hiérarchique à états finis fondé sur les régions, grâce à une optimisation séquentielle, via l'algorithme *Texture Fragmentation and Reconstruction* (TFR). Le TFR permet la modélisation soit des interactions intra-textures, soit des interactions entre textures différentes et donc il résout le problème de la segmentation de manière complètement non supervisée. En outre, il fournit une solution hiérarchique qui peut être interprétée à différentes échelles spatiales en fonction des besoins de l'utilisateur. Différents tests de l'algorithme ont été faits sur des images texturées fournies par le *Prague Texture Segmentation Datagenerator Benchmark* et sur des images de télédétection de forêts fournies par l'Inventaire Forestier National (IFN).

**Mots-clés :** Segmentation de texture, classification, matrices de co-occurrence, modèle structural, chaîne de Markov, synthèse de texture, classification des forêts.

# Contents

**1 Introduction** **6**

**2 TS-MRF model-based segmentation** **7**

**3 Texture Fragmentation and Reconstruction (TFR) algorithm** **9**
3.1 Hierarchical finite-state texture modeling . . . . . 9
3.2 General optimization scheme . . . . . 12
3.3 Color-based clustering (CBC) . . . . . 15
3.4 Spatial-based clustering (SBC) . . . . . 16
3.5 Region merging: the *region gain* . . . . . 17
3.6 Enhanced region gain . . . . . 19

**4 Experimental Results** **20**
4.1 The Prague Texture Segmentation Datagenerator Benchmark . . . . . 20
4.1.1 Performance assessment . . . . . 21
4.1.2 Comparative segmentation algorithms . . . . . 24
4.1.3 Texture mosaic segmentation results . . . . . 25
4.2 Application to remote-sensing images covering forest areas . . . . . 40

**5 Conclusion** **43**

# Glossary

## Acronyms

| | |
|---|---|
| AR3D | 3-D Auto Regressive model |
| CBC | Color-Based Clustering |
| CBIR | Content-Based Image Retrieval |
| EDISON | Edge Detection and Image SegmentatiON system |
| EM | Expectation-Maximization estimation algorithm |
| GM | Gaussian Mixture |
| GMRF | Gauss-MRF model |
| H-RAG | Hierarchical Region Adjacency Graph |
| KLD | Kullback-Leibler Divergence |
| MDL | Minimum Description Length validation criterion |
| MRF | Markov Random Field |
| PCA | Principal Component Analysis |
| RAG | Region Adjacency Graph |
| SBC | Spatial-Based Clustering |
| TFR | Texture Fragmentation and Reconstruction algorithm |
| TFR+ | TFR with KLD-based region gain |
| TP | Transition Probability |
| TPM | Transition Probability Matrix |
| TS-MRF | Tree-Structured Markov Random Field |

## Symbols

| | |
|---|---|
| $\setminus$ | set minus |
| $\langle\cdot\rangle$ | statistical average |
| $\epsilon_i(\cdot,\cdot)$ | measure error tolerant to refinement |
| $\eta_j(s)$ | neighbor of $s$ in direction $j$ |
| $\Omega$ | quantized color space |
| $B$ | number of spectral bands |
| $C$ | commission error |
| $CA$ | weighted average class accuracy |
| $CC$ | object precision |
| $CI$ | comparison index |
| $CO$ | recall |
| $CS$ | correct detection |
| $D(p\|q)$ | Kullback-Leibler discrimination between $p$ and $q$ |
| $EA$ | mean class accuracy estimate |
| $GCE$ | global consistency error |
| $G_t$ | split gain |
| $\mathcal{G}^i, \mathcal{G}^i_{\mathrm{KL}}$ | gain of region $i$ and its modified version with KLD, respectively |
| $\mathcal{G}^*$ | region gain threshold |
| $I$ | type I error |
| $II$ | type II error |
| $K$ | number of clusters in the $k$-means algorithm |
| $L$ | number of states at the finest representation level |
| $LCE$ | local consistency error |
| $ME$ | missed |
| $MS$ | mapping score |
| $N, \hat{N}$ | number of textures in the image and its estimate |
| $N_c$ | number of connected subregions of the color-quantized region $c$ |
| $NE$ | noise |
| $O$ | omission error |
| $OS$ | over-segmentation |
| $\mathbf{P}, \mathbf{Q}$ | empirical transition probability matrix and its modified version, respectively |
| $\mathcal{R}_i$ | region corresponding to the terminal state $i$ |
| $RM$ | root mean square proportion estimation error |
| $\mathcal{S}$ | image support |
| $\mathcal{S}^c$ | region corresponding to the quantization color $c$ |
| $\mathcal{S}^c_n$ | connected subregion of $\mathcal{S}^c$ |
| $\mathcal{S}_{\omega \xrightarrow{j} \omega'}$ | $\{s \in \mathcal{S}_\omega, \eta_j(s) = \omega'\}$ |
| $T^c_n$ | transition probability matrix associated with region $\mathcal{S}^c_n$ |
| $US$ | under-segmentation |

# 1 Introduction

Image segmentation [4, 7, 13, 23] is a low-level processing which is of critical importance for many applications in several domains, like medical imaging, remote sensing, source coding, and so on. Although it has been widely studied in the last decades in many cases it remains still open, as for textured images, where the spatial interactions may cover long ranges asking for high order complex modeling.

There are a large number of approaches to segmentation, but for the sake of brevity, here we confine ourselves to reviewing only those that have been tested using the same benchmarking system [14] as we use, and which therefore serve as points of comparison. In [12] image blocks are modeled by means of local Gauss Markov Random Fields (GMRF) and the segmentation is performed in the parameter space by assuming an underlying Gaussian Mixture. Similar to the previous, but with an auto-regressive 3-D model (AR3D) in place of the Gauss MRF, is the method presented in [13]. In [8] an approach, namely the JSEG, is presented where segmentation is achieved in two steps: a color quantization followed by a processing of the label map which accounts for spatial interaction. Another method taken in consideration is the segmentation algorithm underlying the content-based image retrieval system *Blobworld* [4]. Here a Gaussian Mixture model is assumed in a feature space, where contrast, anisotropy and polarity are the salient texture descriptors, and the EM algorithm carries out the clustering. Finally, the algorithm presented in [5] (EDISON) combines a region-based approach with a contour-based one, hence balancing the global evidence which characterizes a region-based model with the local information typically dominant in the contour modeling.

In this research report we present a novel method for unsupervised texture segmentation, where the image to be segmented is first discretized and then a hierarchical finite-state region-based model is automatically coupled with the data by means of a sequential optimization scheme, namely the *Texture Fragmentation and Reconstruction* (TFR) algorithm. The model allows to take into account both intra- and inter-texture interactions, and eventually allows to address the segmentation task in a completely unsupervised manner. Moreover, it provides a hierarchical output, such that the user may decide the scale at which the segmentation has to be given on the basis of the specific application.

The TFR is basically composed of two steps. The former focuses on the estimation of the states at the finest level of the hierarchy, and is associated with an image fragmentation, or over-segmentation. The latter deals with the reconstruction of the hierarchy representing the textural interaction at different scales. The reconstruction part is controlled by a measure named *region gain* which accounts for the spatial scale of the corresponding region, and for the "attraction" operated by the neighboring regions on it. In particular two different gains have been defined and compared experimenting with data generated by the benchmark system [14]. A comparison with other methods using the same benchmark was considered as well. Finally, we have considered a very critical remote sensing application, which is the forest segmentation [15], for which traditional color-based segmentation methods usually fail, asking for texture-based algorithms. The data in this case were curtesy of the French National Forest Inventory (IFN).

# 2 TS-MRF model-based segmentation

The texture segmentation algorithm which will be presented later in this research report, makes use of a color-based segmentation as a first step. Among all the possible choices we have selected for this purpose the *tree-structured* MRF (TS-MRF) model-based algorithm presented in [7, 23] and briefly recalled here. This algorithm has several characteristics which are attractive in this context. It uses a MRF prior modeling which helps to regularize elementary regions and improve the robustness with respect to the presence of noise. Furthermore, the data likelihood description is based on a multivariate Gaussian modeling which takes into account the correlation in the color space. Finally, its tree structured formulation, similar to that of the tree-structured vector quantization algorithm [11], speeds up the processing, and ensures convergence to the desired number of clusters. In the following a brief description of the TS-MRF algorithm is given.

A random field $X$ defined on a lattice $\mathcal{S}$ is said to be a MRF with respect to a given neighborhood system if the Markovian property holds for each site $s$. The distribution of a positive MRF can be proved to have a Gibbs form [10], that is:

$$p(x|\theta) = \frac{1}{Z}\exp[-U(x,\theta)], \tag{1}$$

with $U(x,\theta) = \sum_{c\in C} V_c(x_c,\theta)$, where $x$ is the realization of the field $X$, $\theta$ is the set of parameters of the model, the $V_c$ functions are called potentials, $U$ denotes the energy, $Z$ is a normalizing constant that depends on $\theta$, and $c$ indicates a clique of the image. Note that each potential $V_c$ depends only on the values taken on the clique sites $x_c = \{x_s, s \in c\}$ and, therefore, accounts only for local interactions. As a consequence, local dependencies in $X$ can be easily modeled by defining suitable potentials $V_c(\cdot)$. In particular the second order Potts MRF model [2] is considered in this work, where only pairwise cliques are associated with not null potentials, that is:

$$V_c(x_c) = \begin{cases} \beta & \text{if } x_p \neq x_q, \quad p,q \in c \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $\beta > 0$ is the model parameter.

Due to the inherent high complexity of this model, which can be optimized by stochastic relaxation algorithms or other similar procedures, a faster algorithm for unsupervised image segmentation, which is based on "tree-structured" MRF modeling, has been developed in [7].

Let us now consider a $K$-class image segmentation problem and model the unknown label map with a random field $X$ defined on the lattice $\mathcal{S}$ of the image $y$ to be segmented. Such a problem can be viewed as a nesting of several segmentation problems of reduced complexity. Given a binary tree structure with $K$ terminal leaves, each associated with one of the $K$ class/region to be singled out, there will be $K-1$ internal nodes, each representing the merging region of the descendant leaves/regions. Equivalently, each of such larger region represents the irregular[1] (except for the root node) support domain for its children regions, which correspond either to terminal classes or to merging classes. Actually, in the TS-MRF unsupervised algorithm such a structure and the corresponding image partitioning are recursively singled out by top-down induction, where each step is a

[1] Not rectangular.

binary region split (associated with a node split) based on a local binary MRF like the Potts model presented above, with $K = 2$, and defined on a proper, irregular, lattice shaped by the ancestor splits.

The growth of the tree, that is the choice of the node to be split from time to time is controlled by a test local to each node, namely the split gain, which accounts for the likelihood of the hypothesis of split of the corresponding node with respect to the hypothesis of non split of the same node, on the basis of the observed local test segmentation (see [7] for further details). Also, when all such tests on the current terminal leaves indicate to not split, the tree growth is stopped and the number of classes is automatically given by the number of leaves.

Indeed, since we will use the TS-MRF as part of a more complex tool for texture segmentation, the cluster validation problem has not to be solved by the TS-MRF itself which, eventually, will simply over-segment the image. For this reason we define here a different split gain, to not be meant as hypothesis test but only as indicator of the total distortion decrease obtained by fitting separately the two regions of a split with local models. Let us label $t$ the node/region to be split and be $r$ and $l$ the right and left children respectively, therefore the split gain is defined as:

$$G_t(x^t) = \frac{p_r(y^r)p_l(y^l)}{p_t(y^t)}, \tag{3}$$

where $x^t$ is the test segmentation[2], $y^i$ is the image portion associated with node $i$, and $p_i(\cdot)$ is the best matching Gaussian distribution for the data $y^i$. Eventually, the TS-MRF algorithm used here differs form the version presented in [7] in the way how the tree growth is controlled. In particular the split gain defined above does not contain an additional term, namely the prior probability $p(x^t)$ (see [7]), which accounts for the shape regularity of the children regions but is unnecessary in this context. Also the growth is stopped when the required number of classes is reached and not when all gains are under a threshold.

[2] A region segmentation is first computed and, hence, on the basis of the corresponding split gain is validated or not.

# 3 Texture Fragmentation and Reconstruction (TFR) algorithm

The proposed segmentation method, hereafter referred to as *Texture Fragmentation and Reconstruction* (TFR) algorithm, is based on a hierarchical finite-state modeling of the image textures, and is optimized by means of a split-and-merge procedure. The former (top-down) step aims at decomposing the image to be segmented (then, also each of its textures) in a sufficiently large number of components, hence it over-segments the image. The latter (bottom-up) step aims at associating recursively the texture components previously extracted so that each different texture is properly reconstructed. The process provides us at the end with a hierarchical segmentation map, as *structured* textures can be revealed at different scales. By structured texture we mean a spatial pattern whose interaction range, i.e. the order, is not clearly bounded but can be regarded as the superposition of simpler bounded-range interactions. An image pattern containing both macro and micro textural interactions is such an example.

In general, a complex scenario can be also regarded as a single texture at the coarsest scale, and, in a sense, the same cluster validation problem[3] does not make sense, i.e. it is an ill-posed problem, if the scale is not fixed somehow. As an example, consider the front of a building with an array of windows. At a finest scale one likely can distinguish the *glasses*, the *frames* of the windows, and the *walls*. Then, at a coarser scale, frames and glasses can be considered as a unique texture (*window*), since they are strongly related spatially, while at the coarsest scale window and walls, which also relate to each other but with longer range spatial interactions, merge in the *building* texture.

The ill-positioning of the cluster validation problem is very common in many computer vision applications, and, being aware of its strict correlation with the spatial scale, herein we will provide a method which gives a hierarchical segmentation rather than a single segmentation with an estimated (somewhat "unreliable") number of regions. By doing so, we get a scale-dependent interpretation of the image, represented by a set of nested segmentations which can be associated with a tree structure where each of its pruning corresponds to a possible segmentation.

## 3.1 Hierarchical finite-state texture modeling

In order to achieve the goal discussed above, we resort to an underlying *hierarchical*, *discrete* and *region-based* modeling of the textures. As a consequence, a first basic step is the transformation from the continuous space of the image $\mathbb{R}^{B\times|\mathcal{S}|}$, with $B$ spectral bands and support lattice $\mathcal{S}$, to a discrete one $\Omega^{|\mathcal{S}|}$, where $\Omega$ is a finite set. Such a process can be either just a color quantization directly applied to the original image or, more generally, a clustering in any feature space where the image can be projected first. In both cases, the cardinality of $\Omega$ fixes a "resolution" of the model, that in the first case corresponds to a color resolution. Indeed, as it will be clearer later, since this is a region-based method, the higher the cardinality of $\Omega$, the smaller the size of the basic image region elements, hence, the model resolution is also meant as a spatial one.

A natural question is then if the model resolution has any limit, i.e. if the cardinality of $\Omega$ can be chosen arbitrarily large. Though intuitively the answer is negative, it needs some further details about the model in order to be motivated. However, regardless of the motivations, one could doubt

[3] That is, the identification of the number of textures in the image.

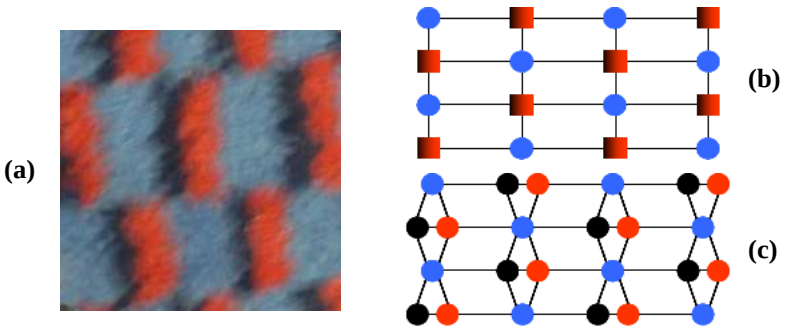



**Figure 1: Hierarchical Region Adjacency Graph (H-RAG) texture representation: textile pattern (a), and two levels of the H-RAG (b)-(c).**

about the texture description capability of the model due to the initial discretization of the image. Actually, while this could be a rather serious limit in a synthesis problem, it is not that critical in an analysis problem like the segmentation, and especially in an unsupervised setting where robustness, rather then precision, is the most relevant issue.

Let us focus now on the region-hierarchical modeling aspects. A very common graph representation of a set of regions which accounts for their relative spatial positioning, is the *Region Adjacency Graph* (RAG), that is a non-oriented graph where nodes are associated with regions, and any connection between two nodes indicates the adjacency of the two corresponding regions. As an example consider the texture of Fig.1 (a). It is immediate to figure a three-level quantized images, with codes *blue*, *black* and *red*. Also, consider as elementary regions all the connected areas with the same code. For such an image partitioning in regions, the corresponding RAG is depicted in (c), where each node represents a connected region and its color recalls the associated color-code. Furthermore, notice that the graph is periodic because of the periodical behavior of the texture. Likewise, in (b) is shown the RAG for a two-level discretization of the same texture where the black and the red regions are merged in a mixing class. More generally, given a set of nested image partitions, we can associate with it a cascade of RAGs, namely the H-RAG (Hierarchical RAG), or the HARAG (Hierarchical Attributed RAG) if the nodes are also featured [9].

Such a modeling have been used mostly for content-based image retrieval applications (CBIR) [9, 24], where the regions are spatially related without any stationarity and the goal is to find a subgraph (hence, an image area) which well fits with the structural definition of the searched object. Instead, for texture modeling some stationarity is expected, as shown in the simple case of Fig.1 where a strong periodicity is well visible and represented by its H-RAG as well. Likewise, an aperiodic texture would have an associated H-RAG which reflects its statistical properties.

A compact, and actually enhanced, representation of the H-RAG for a stationary texture is possible by means of a hierarchical finite-state modeling, as clarified by Fig.2 w.r.t. the above example. Let us consider the eight main spatial directions (north, north-east, east, etc...) and for each of them consider the intra- and inter-region transition probabilities (TPs). Hence, for each partitioning degree, we can define a set of eight state diagrams. For example, in (a) is shown the state diagram

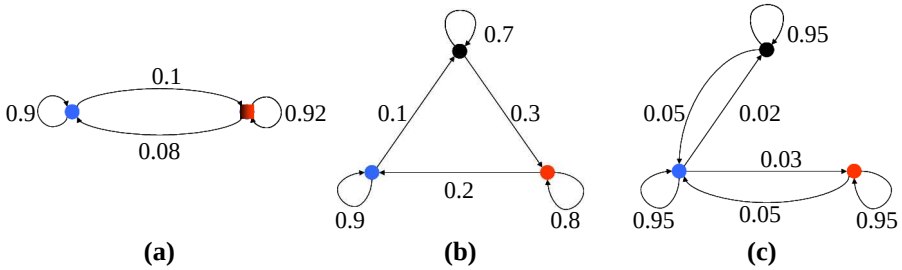



**Figure 2: Finite-state modeling associated with the texture of Fig.1. Two-state model: east direction (a). Three-state model: east (b) and south (c) directions.**

corresponding the the east direction when the texture is reduced to only two states, while in (b) and (c) are plotted diagrams w.r.t. the three-state partition, for the east and south directions, respectively. This representation is enhanced since the H-RAG only reveals the adjacency, regardless of its weight and directionality. Approximated TPs are indicated on the graphs just to give an idea of their relationship with the visual appearance of the texture. To be more precise, in talking about TPs we have to specify the spatial step size. In the following, we simply refer to the pixel size, since we assume to move on a pixel-by-pixel basis according to a 8-connected neighborhood system.

The interpretation of this graphical representation is rather immediate. First, note that the intra-region TPs account for the shape of the texture components at any region-scale. Consider, for example, the blue patches that regularly occur in the texture sample. Due to their rectangular shape, the associated intra-region TP in the vertical direction (c) is larger then the horizontal one (b). Furthermore, the remaining, inter-region, TPs account for the spatial context, that is, the relative occurrence and positioning of the neighboring regions. For the transitions between states it is very common to refer to the conditional probabilities given that the state changes, which are scale invariant in this particular case. Finally, observe that once the TPs are known at a given level in the hierarchy, then they are automatically obtained also for the above levels and, eventually, one only has to estimate these attributes at the finest level.

From the segmentation point of view, it is necessary to move from the global characterizations like the H-RAG or the finite-state models to a local characterization at the region level. According to the modeling given above, the natural attributes to associate with any region are its color code and its local TPs.

To be more formal, let us introduce a few notations. Given a texture image $Y \in \mathbb{R}^{B\times|\mathcal{S}|}$, let $Z \in \Omega^{|\mathcal{S}|}$ be its finest-level discrete-state approximation, where $\Omega$ is the set of all possible states. At first glance, it could seem that these states are just the quantization colors, which is actually not necessary true since the same color may occur in a texture according to different configurations of TP, and then different states are defined. Hence, let $\mathcal{S}_\omega \subseteq \mathcal{S} = \{s \in \mathcal{S}, Z_s = \omega\}$ be the set of image pixels with state $\omega \in \Omega$, then the associated $|\Omega| \times 8$ transition probability matrix (TPM), $\mathbf{P}_\omega$, is

computed as

$$\mathbf{P}_\omega(\omega', j) = \frac{\#\text{ of pixels } s \in \mathcal{S}_\omega \text{ such that } \eta_j(s) \in \mathcal{S}_{\omega'}}{\#\text{ of pixels } s \in \mathcal{S}_\omega} \triangleq \frac{|\mathcal{S}_{\omega \overset{j}{\rightarrow} \omega'}|}{|\mathcal{S}_\omega|} \qquad \forall \omega' \in \Omega, \;\; 1 \leq j \leq 8, \tag{4}$$

where $\eta_j(s)$, is the neighbor of pixel $s$ in position (direction) $j$. Eventually, it $\mathcal{T}$ indicates the tree structure which represents a region hierarchy, according to our model a texture is defined by the triple $(\Omega, \mathcal{P}, \mathcal{T})$, with $\mathcal{P} = \{\mathbf{P}_\omega\}_{\omega \in \Omega}$.

Finally, a local characterization is easily derived by dealing with the elements of the partition of each $\mathcal{S}_\omega$ in its connected subregions $\mathcal{S}_\omega^n$, $n = 1, \ldots, N_\omega$. In fact, any such region has an own TPM which differs from the global one, i.e.

$$\mathbf{P}_\omega^n(\omega', j) = \frac{|\mathcal{S}_{\omega \overset{j}{\rightarrow} \omega'}^n|}{|\mathcal{S}_\omega^n|} \qquad \forall \omega' \in \Omega, \;\; 1 \leq j \leq 8, \tag{5}$$

but is related to, as can be easily shown, by the weighted average:

$$\mathbf{P}_\omega = \frac{1}{|\mathcal{S}_\omega|} \sum_{n=1}^{N_\omega} |\mathcal{S}_\omega^n| \mathbf{P}_\omega^n. \tag{6}$$

This is just because we are considering a union of regions. Likewise, once the hierarchy $\mathcal{T}$ is given, since the intermediate states are obtained by merging the terminals, similar linear combinations allow us to derive the TPMs at any coarser level.

## 3.2 General optimization scheme

In the previous section we exploited a discrete and hierarchical texture modeling, regardless of any particular application. Here, we will focus on the unsupervised segmentation problem and propose an optimization algorithm for this case. A few comments have to be given about the problem setting. Since we are assuming an unsupervised context, then we do not know *a priori* how many and what kind of textures may be found in the image to be segmented. Also, we cannot even restrict the variety of image patterns of interest to a narrow-domain, since we want to focus explicitly on the broad-domain case, being aware that eventually alternative and particularized optimizations can be derived, for sure, if any restriction can be assumed.

As a consequence, items to be estimated are the number of textures and, for each texture, the terminal[4] states, with corresponding TPMs, and the hierarchical relationships by means of which intermediate states with associated attributes are derived by combining the terminals.

The determination of the number of textures of a given image, classically referred to as *cluster validation problem*, is strictly related to that of finding the internal structure of each single texture, especially when long range interactions are present in the textures. The example in Fig.3 clarifies this observation. The image (a) is composed of two textures whose representative states at the coarsest level are indicated by $w$ and $z$. In a H-RAG representation of a texture, the coarser level corresponds

[4]Corresponding to the finest level in the hierarchal modeling.

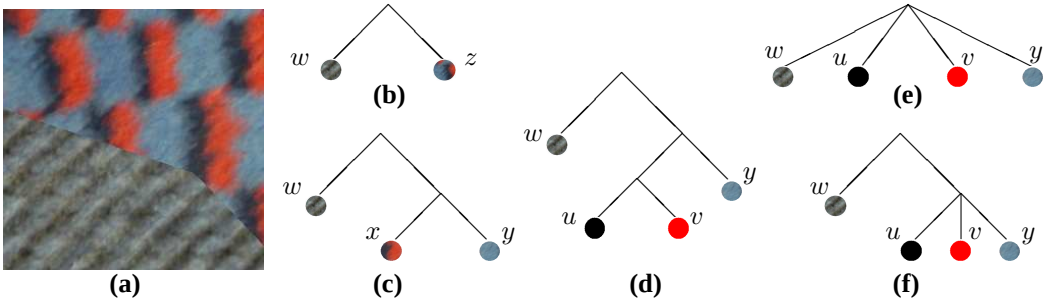



**Figure 3: Image structure ambiguity. A texture mosaic (a) and several binary (b)-(d) and non-binary (e)-(f) hierarchical modelings.**

to the root of the hierarchical structure and, eventually, is associated with the texture as whole. Moreover, if an image contains more than one texture, than the whole image can be considered itself as a texture whose hierarchical structure is simply obtained by relating the marginal substructures until a root node is reached. In this sense, the structures shown in Fig.3 (limited in depth for sake of simplicity) represent both intra- and inter-texture dependencies. A first observation to make is about the ill-positioning of the cluster validation problem. Indeed, for such an image a human observer could guess that the textures are actually four and not just two, and hence we can expect that for a computer these data are even more confusing. In terms of hierarchical representation, at the top level the two different cases correspond to (b) and (e).

Eventually, this ill-posed problem needs to be regarded from a different point of view in order to be reasonably solved. So, rather than really finding the exact number of textures we will provide a *hierarchical segmentation*, that is a set of nested segmentations (hence, coupled with a tree structure) where a single segmentation corresponds to a pruning of the full tree. By doing so, as final product we only have to provide the finest segmentation and the region hierarchy tree which univocally determines any coarser segmentation, leaving to the user the choice of the pruning (i.e. the number of classes) which better fits with the subsequent application.

Indeed, this is rather reasonable since in most of the applications, segmentation is just an early processing which precedes some other ones. As an example, if we consider data compression and use a region-based technique, then segmentation is required as a first step and then single regions extracted are optimally coded. If we refer to the image of Fig.3 (a), then we may reasonably expect the 4-class partition to be preferable to the 2-class one. In terms of hierarchical representation for the provided segmentation, then we may expect a structure which figures at the top levels like (f). As it can be seen, the full structure (f) corresponds to the desired 4-class segmentation, while a pruning of it would provide us with a structure like (b) which gives the 2-class partition. Notice that a flat structure like (e) would not contain any substructure which corresponds to the 2-class segmentation.

Finally, let us limit our attention to the case of "binary" structures. Comparing (d) and (f), we can see at first glance that (d) is an approximation of (f) when the binary constraint is assumed. Indeed, from our point of view (d) is reacher than (f), since it contains the same segmentations as (f), plus

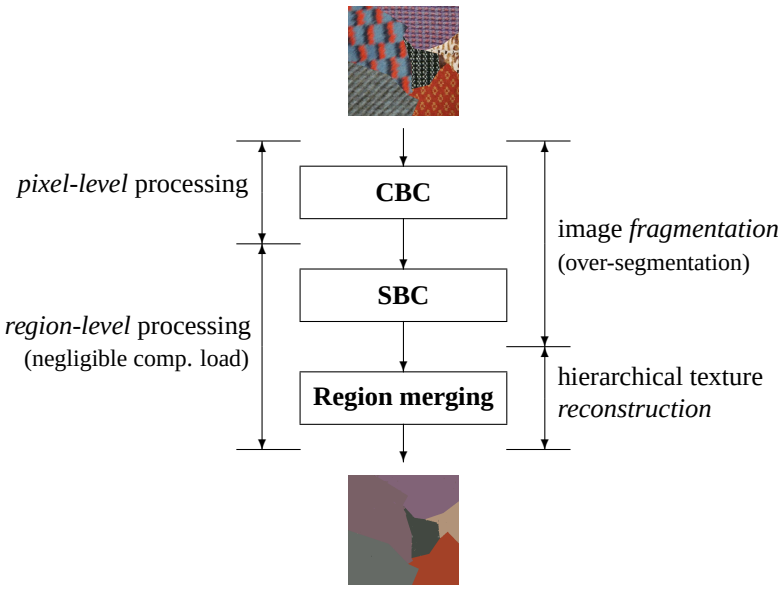



**Figure 4: Flow chart of the Texture Fragmentation and Reconstruction (TFR) algorithm. (CBC: *color-based* clustering; SBC: *spatial-based* clustering)**

one more, a 3-class segmentation. This is simply due to the presence of one more internal node in the structure which increases the number of possible prunings, i.e. image interpretations. Moreover, the binary constraint simplifies the search of the tree structure to be associated with the segmentation because it limits the number of possible structures to investigate. From the above considerations, in the following we only deal with binary hierarchies.

Let us turn now back to the optimization problem. As we re-formulate the cluster validation problem, it is no more a critical part of the optimization. Therefore, it is partially solved when the global H-RAG is determined, and can be completed by any application-dependent criterion to be applied on the final hierarchical segmentation provided which, actually, has just to identify an optimal pruning.

Therefore, we can neglect the number of textures, then what remains to be estimated is a "reasonable" number[5] of terminal states and the global hierarchical tree. The optimization scheme we propose (see Fig.4) is quite simple but effective, as it will be shown by the experimental tests. Notice first that the extraction of the states and the hierarchy are performed in two separate and sequential steps. The former, composed by the blocks CBC (Color-Based Clustering) and SBC (Spatial-Based Clustering), dealing with the estimate of the states, the latter focused on the hierarchy. The split of the former in two parts is justified by the fact that a state in our model is characterized by means of

[5]The term "reasonable" reminds that by choosing that number we are fixing somehow the resolution of the algorithm.

the region response, either in the original color space or in a transformed one, and the TPM which accounts for shape and neighborhood of the regions. Furthermore, while the color response is a pixel-wise feature, the TPM is a region-wise characteristic. For these reasons we propose to process color and spatial information independently in a sequential way. The CBC block will create a certain number of *partial*-states, namely the color-states, discriminated just by the color, and then the SBC will further split them on the basis of the region-based features. Since SBC, as well as the subsequent region merging, works at region-level (it handles the elementary connected regions created by CBC) the associated computational load is negligible w.r.t. the CBC burden. Region merging, or state merging, is nothing but a sequential binary combination of the states driven by a specific parameter which accounts for the mutual spatial relationships among the states.

Let us detail now the single steps of the TFR algorithm.

## 3.3 Color-based clustering (CBC)

This segmentation step is critical in the sense that color information is handled only here and regardless of spatial interaction. Since one has to extract a certain number of color-states, what has to be performed is a color-based segmentation or clustering. Hence, CBC can be approached by using one of the many well known methods for this kind of problem, like vector quantization [11], low order MRF-based algorithms [10, 17], and so on.

As CBC we have used the tree-structured MRF (TS-MRF) algorithm [7]. This algorithm has several features which are attractive in this context. It uses a MRF prior modeling which helps to regularize elementary regions and improves the robustness with respect to the presence of noise. As matter of fact, it gives larger uniform regions, whose characterization in terms of TPM is more reliable compared to regions whose size approaches the pixel's one. Furthermore, the data likelihood description is based on a multivariate Gaussian modeling which takes into account the correlation in the color space. Finally, its tree structured formulation, similar to that of the tree-structured vector quantization algorithm [11], speeds up the processing, and allows for adaptively choosing the color-cluster to be split in a recursive process. Such a feature may be used to make uniform the size of the region elements provided by the color clustering, as to ensure a uniform spatial resolution of the method.

The number of regions to be singled out needs to be either fixed a priori, on the basis of some heuristic rule, or estimated according to some cluster validation criterion. To this end, a trade-off should be taken into account. In fact, a high number of regions produced at this stage, although would help to preserve image details, will produce too small elementary regions whose subsequent spatial characterization for next processing would be less reliable. Also, eventual longer range spatial interactions of a given texture would need more steps to propagate in the finite-state modeling, and hence ask for a more complex optimization process to be recognized.

Being aware of these aspects, we decided to use a simple heuristic rule, to be refined in future work, as at present other aspects of the algorithm seem to be more critical and necessitate to be addressed first. On the basis of our experimental observations, we found the double of the maximum number of textures expected in the image to be a reasonable choice for the number of color clusters to be extracted. This can be intuitively justified by the fact that any non-trivial texture has at least

two modes in the color space. Hence, we are ensuring that, in average, we have at least two colors per texture.

### 3.4 Spatial-based clustering (SBC)

In the previous section we described how a certain number of partially-defined states, or color-states, are derived from the image by CBC discriminating only w.r.t. color. The spatial-based clustering (SBC) then focuses on the further split of these states by using the contextual information contained in the TPMs defined in Section 3.1.

In principle, this splitting process should be carried out simultaneously for all the color-states, as the characterization of the fully-defined states would not refer to partially-defined states. In other words, while computing the TPM of a region of a given color, one should refer to fully-defined states, requiring the other color-states to be already split. Therefore, it is clear that the optimization problem may be addressed iteratively by alternating state estimation and region labeling. However two main concerns about this solution have to be noticed. The first one is that we have no guarantees about the convergence of such an iterative process. The second one is about the reliability of the characterization of the states. While the former is easily understood and perhaps can be addressed in practice, the latter is more critical and needs to be further clarified.

For this purpose consider the example below. Let 24 be the number of color-states computed by CBC, and suppose the CBC output map to have around $10^4$ connected regions, i.e. image elements to be characterized and clustered. Also let 12 be the average number of offspring states per color-state. Hence the total number of states is $24 \times 12 = 288$, and the TPMs will contain $288 \times 8 = 2304$ elements with an high degree of sparseness. This means that we should perform a clustering in a space where the ratio between the number of elements and the dimensionality of the space is around $10^4/2304 \approx 4$, which is clearly not enough and, then, a considerable feature reduction is needed. Furthermore, an eventual feature reduction should be iterated as well as the clustering, causing huge computations and making even more critical the convergence of the process.

For these reasons we implemented a different optimization scheme which does not require a joint split of color-states. In practice we consider an approximate characterization of the image elements, by simply referring to the partially-defined states.[6] In this way we factorize the optimization process, since the splits of the color-states are mutually independent. Nevertheless, the dimensionality of the feature space is still large for a reliable characterization, and then we use to a principal component analysis (PCA) to reduce it.

In particular a single split, which is a clustering of connected regions with the same color-state, say $\omega$, is carried out as follows. For each connected element $n \in \{1, \ldots, N_\omega\}$, the corresponding TPM $\mathbf{P}_\omega^n$ is computed by Eq.5. Experimentally we found more effective the use of a logarithmic transformation $\mathbf{Q}_\omega^n$ defined as:

$$\mathbf{Q}_\omega^n(\omega', j) \triangleq \begin{cases} \log[1 - \mathbf{Q}_\omega^n(\omega', j)], & \omega' = \omega \\ \log[\mathbf{Q}_\omega^n(\omega', j)/(1 - \mathbf{Q}_\omega^n(\omega, j))], & \omega' \neq \omega. \end{cases} \tag{7}$$

[6] W.r.t. the previous example, only $24 \times 8$ TPMs are now involved.

Apart the use of the logarithm, notice that for $\omega' = \omega$ this corresponds to replace the probability of keeping the current state with its complement (i.e. probability to leave the state). When $\omega' \neq \omega$, that is the case when state $\omega$ is left for $\omega'$, we just conditioned w.r.t. the event "$\omega$ is left".

Now for each fixed $\omega'$ consider the 8-dimensional row vectors $\mathbf{Q}_\omega^n(\omega', \cdot)$, $1 \leq n \leq N_\omega$, and reduce them to scalar values by PCA, i.e. by taking the projecting on the largest eigenvector. Roughly speaking we are just considering an average behavior w.r.t. the spatial directions, where those which are more "informative" are weighted more. By doing so independently for each row $\omega'$, we reduce the matrices $\mathbf{Q}_\omega^n$ to a $|\Omega|$-dimensional column vector, which is then further reduced by PCA, as to ensure a smaller dimensionality. In particular the number of meaningful components is automatically chosen to keep 75% of the energy. This second PCA is justified by the fact that some neighbor states actually occur rarely and are due to the noise. Therefore a PCA improves further the robustness of the clustering.

Finally, the clustering is performed by applying a $k$-means algorithm in this feature subspace where data are no longer as sparse as in the full space spanned by the TPMs. The number $K$ of subclusters per each color-state split is fixed a priori, since its exact computation is not necessary. In particular, $K$ must be large enough to avoid under-clustering, that eventually could cause the merging of different textures during the subsequent region merging step. On the other hand, a too large $K$, hence an over production of states, may cause a single texture to be split afterwards. According to our experience, a good compromise is to fix $K$ to the expected number of texture in the image, or to its upper bound if this is the only known information.

## 3.5 Region merging: the *region gain*

The result of the sequence of steps described above (CBC and SBC), is a partition of the image in regions, each of them corresponding to a state defined in terms of color, shape and spatial context. According to the hierarchical modeling formulated above (see Section 3.1), these are the terminal states (finest scale) which have now to be related until all collapse in the macro state associated with the hierarchy root, i.e. with the whole image (coarsest scale), which corresponds to a recursive region merging. The aim of this process is to collect together states, i.e. regions, that belong to the same texture as to obtain the texture identification and segmentation. Now, given that a single state may interact with both the states of the same texture and those of a neighboring one, what is important is to privilege the merging of states strongly correlated, as they likely belong to the same texture. In this way the intra-texture mergings will be privileged w.r.t. the inter-texture ones that, eventually, will appear at the top levels of the hierarchy. In particular, a correct structure identification requires all intra-texture mergings to be performed before any inter-texture one. If we achieve this, then all marginal texture models are enclosed as non-overlapped substructures in the overall hierarchical model, hence a separation of them is possible by simply stopping the merging process when all the intra-texture mergings are done.

As already discussed above, we focus in this report on binary hierarchies in order to reduce the complexity, as well as to benefit of the greater flexibility of a binary solution for the purpose of representation. Therefore the structure building can be performed by means of a recursive sequence of binary mergings driven by a test parameter. Due to the above remark, we have properly defined a

test, namely the *region gain*, which is defined in the following. At any given step $t$ of the recursive merging sequence, let $\Omega^{(t)}$ be the current set of states which are candidates for the next merging. Then $\Omega^{(0)}$ is the set of terminal states, while $\Omega^{(L-1)}$ is just the root state, with $L = |\Omega^{(0)}|$. Now, for each state $i \in \Omega^{(t)}$, corresponding to region $\mathcal{R}_i$, the region gain is defined as:

$$\mathcal{G}^i = \frac{p(s \in \mathcal{R}_i)}{\max_{j \neq i} p(r \in \mathcal{R}_j | s \in \mathcal{R}_i)} \tag{8}$$

$$= p(s \in \mathcal{R}_i) \cdot \frac{1}{p(r \notin \mathcal{R}_i | s \in \mathcal{R}_i)} \cdot \frac{p(r \notin \mathcal{R}_i | s \in \mathcal{R}_i)}{\max_{j \neq i} p(r \in \mathcal{R}_j | s \in \mathcal{R}_i)} \tag{9}$$

where $s$ is an image site and $r$ is any of the eight neighbors of $s$.

Notice that the test parameter is not a measure of similarity between states but a measure (inverse) of the migration speed of state $i$ toward other states, in particular the most attractive state, $j^* = \arg\max_{j \neq i} p(r \in \mathcal{R}_j | s \in \mathcal{R}_i)$. In fact, it is a ratio between the area corresponding to state $i$, which can be considered as a weight, and the probability to leave state $i$ for $j^*$ (Eq.8).

Eventually, at current time $t$ the weakest state $i$, that is associated to the minimum gain, is absorbed by the state $j^*$ that attracts it the most. Then the merging process is iterated until the root of the hierarchy is reached.

The factorization in Eq.9, allows us to give an easy interpretation of the gain, as the spatial scale dependence of the output hierarchy. Indeed, the gain of a state is also an indicator of the scale of its corresponding region, and consistently it typically increases when merging states and associated areas. The first factor is clearly related to the scale since it is proportional to the area, as well as the second, which indicates the compactness of the region. In fact, the larger the probability to leave the state (denominator), the larger the region perimeter, then the less compact the region is distributed. The third factor, which is just the inverse of the occurrence of the nearest state given that the state is left, is instead related to the spatial context rather than to the scale. For a fixed region, in fact, the presence of a dominant neighbor rather than several equally occurring ones, indicates a clear relationship between two states which likely have to be merged.

Recall that once the complete sequence of merging is defined then a nested hierarchical segmentation is given. Therefore, any user may select the proper segmentation he/she needs depending on the purpose. Sometimes, instead, the user has additional information that may drive the validation in a post-processing. Nevertheless, this problem can be addressed just in terms of scale, since the region gain is directly related to it and it is sufficient to give a gain threshold. In this sense, fixing the scale corresponds to take a reference region with fixed area (say $\alpha \times |\mathcal{S}|$), shape (say a square) and context (say only one surrounding neighbor). With these constraints the given region will have approximatively a gain equal to:

$$\mathcal{G}^* = \frac{8\alpha(\sqrt{\alpha|\mathcal{S}|} - 2)^2}{12\sqrt{\alpha|\mathcal{S}|}}, \tag{10}$$

which can be used as a threshold through the specification of $\alpha$.

Concerning the computational load of the merging process, it is easy to recognize that it is very light, since one only has to compute the TPMs for the terminal states provided by SBC, and then

keep an ordered list of the gains of the current states while storing the structure defined by the merge sequence. Also, the TPMs of merging states are just combinations of other TPMs (hence no computation at pixel level), and the gains are derived from the TPMs as well.

## 3.6 Enhanced region gain

The region gain defined above measures how likely the region is itself the support of a texture w.r.t. the hypothesis that it is just a part of a larger support. When the gain is low we let the region be absorbed from the "nearest", as to obtain a larger one. By definition of the region gain, the "nearest" means the neighboring region that shares the largest boundary with the given region. Although this criterion has provided good results, there are certain cases where it fails. Indeed, the presence of noise may increase the length of the boundary between two regions and make them closer according to the gain definition. This problem occurs often because of the boundary fragmentation phenomena caused by color quantization during the CBC step.

In order to reinforce the measure, as to improve the robustness, we considered not only the degree of contact between regions but also their spatial distribution similarity. To do so we have introduced an additional term in the gain, that is the Kullback-Leibler divergence (KLD) [18]. The KLD between two distribution, $p$ and $q$, is defined as:

$$D(p\|q) \triangleq \left\langle \log \frac{p(\mathbf{x})}{q(\mathbf{x})} \right\rangle_p = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x}, \tag{11}$$

where $\langle \cdot \rangle_p$ is the statistical average according to the distribution $p$. Since $D(p\|q)$ is the average log-likelihood ratio between $p$ and $q$, then it is a measure of the inefficiency of assuming $q$ in place of $p$. Hence it is well adapted to describe how much two objects are close w.r.t. their spatial locations. In particular, named $q_i(\mathbf{x})$ the distribution of the spatial location of region $i$, where $\mathbf{x}$ is the 2-D spatial position, then the modified region gain $\mathcal{G}^i_{\mathrm{KL}}$ of state $i$ is defined by:

$$\log \mathcal{G}^i_{\mathrm{KL}} \triangleq \min_{j \neq i} \left\{ \log \frac{p(s \in \mathcal{R}_i)}{p(r \in \mathcal{R}_j | s \in \mathcal{R}_i)} + D(q_i \| q_j) \right\} \tag{12}$$

$$= \min_{j \neq i} \left\{ \log \frac{p(s \in \mathcal{R}_i)}{p(r \in \mathcal{R}_j | s \in \mathcal{R}_i)} + \left\langle \log \frac{q_i(\mathbf{x})}{q_j(\mathbf{x})} \right\rangle_{q_i} \right\}, \tag{13}$$

where we referred to the logarithmic formulation to properly combine the previous gain with the KLD term. Notice that by removing the KLD term the gain reduces to the original one.

The computation of the KLD is in general quite difficult for most of the distributions, and in a few cases admits a closed form. One such case is that of two Gaussian distributions $p$ and $q$ for which the divergence is given by [22]:

$$D(p\|q) = \frac{1}{2} \left( \log \frac{|\Sigma_q|}{|\Sigma_p|} + \mathrm{tr}(\Sigma_q^{-1}\Sigma_p) + (\mu_p - \mu_q)^T \Sigma_q^{-1} (\mu_p - \mu_q) - d \right) \tag{14}$$

where $p \sim \mathcal{N}(\mu_p, \Sigma_p)$, $q \sim \mathcal{N}(\mu_q, \Sigma_q)$ and $d = 2$ is the distribution dimensionality. Due to its simplicity, the above modeling has been considered for comparing the spatial distribution of different regions, and used for the new region gain (Eq.12).

# 4 Experimental Results

## 4.1 The Prague Texture Segmentation Datagenerator Benchmark

The Prague texture segmentation benchmark [14], developed by the *Pattern Recognition* research group of the Institute of Information Theory and Automation, Czech Academy of Sciences, has a two-fold objective:

- to mutually compare and rank different texture segmenters (supervised or unsupervised),
- to support new segmentation and classification method developments.

In particular the server allows:

- to obtain customized experimental texture mosaics and their corresponding ground truth,
- to obtain the benchmark texture mosaic sets with their corresponding ground truth,
- to evaluate any working segmentation result and compare it to the results obtained by the state-of-the-art algorithms,
- to include any algorithm (reference, abstract and benchmark results) into the benchmark database,
- to check single mosaic evaluation details (criteria values and resulted thematic maps),
- to rank segmentation algorithms according to the most common benchmark criteria,
- to obtain LaTeX coded resulting criteria tables.

The computer generated texture mosaics and benchmarks provided by the server are composed of the following texture types:

- monospectral textures,
- multispectral textures,
- BTF (bidirectional texture function) textures,
- rotation invariant texture set (work in progress),
- scale invariant texture set (work in progress).

Furthermore, all generated texture mosaics can be corrupted with additive noise, and training sets are supplied in a supervised mode.

### 4.1.1 Performance assessment

The benchmark server provides a comparative analysis of all the results uploaded by users according to several accuracy indicators which are grouped in three main categories as detailed in the following.

Region-based criteria

The region-based criteria [16] mutually compare the machine segmented regions $\mathcal{R}_i, i = 1, \ldots, M$ with the correct ground truth regions $\bar{\mathcal{R}}_j, j = 1, \ldots, N$. The regions overlap acceptance is controlled by the threshold $k = 0.75$ ($0.5 < k < 1$). Single region-based criteria are defined as follows:

- $CS$ (correct detection): $[\mathcal{R}_m; \bar{\mathcal{R}}_n]$ iff
  - (i) $\mathrm{card}\{\mathcal{R}_m \cap \bar{\mathcal{R}}_n\} \geq k\,\mathrm{card}\{\mathcal{R}_m\}$
  - (ii) $\mathrm{card}\{\mathcal{R}_m \cap \bar{\mathcal{R}}_n\} \geq k\,\mathrm{card}\{\bar{\mathcal{R}}_n\}$
- $OS$ (over-segmentation): $[\mathcal{R}_{m1}, \ldots, \mathcal{R}_{mx}; \bar{\mathcal{R}}_n]$, $2 \leq x \leq M$ iff
  - (i) $\forall i \in \{1, \ldots, x\}$, $\mathrm{card}\{\mathcal{R}_{mi} \cap \bar{\mathcal{R}}_n\} \geq k\,\mathrm{card}\{\mathcal{R}_{mi}\}$
  - (ii) $\sum_{i=1}^{x} \mathrm{card}\{\mathcal{R}_{mi} \cap \bar{\mathcal{R}}_n\} \geq k\,\mathrm{card}\{\bar{\mathcal{R}}_n\}$
- $US$ (under-segmentation): $[\mathcal{R}_m; \bar{\mathcal{R}}_{n1}, \ldots, \bar{\mathcal{R}}_{nx}]$, $2 \leq x \leq N$ iff
  - (i) $\sum_{i=1}^{x} \mathrm{card}\{\mathcal{R}_m \cap \bar{\mathcal{R}}_{ni}\} \geq k\,\mathrm{card}\{\mathcal{R}_m\}$
  - (ii) $\forall i \in \{1, \ldots, x\}$, $\mathrm{card}\{\mathcal{R}_m \cap \bar{\mathcal{R}}_{ni}\} \geq k\,\mathrm{card}\{\bar{\mathcal{R}}_{ni}\}$
- $ME$ (missed): $[\bar{\mathcal{R}}_n]$ iff
  - (i) $\bar{\mathcal{R}}_n \notin$ correct detection
  - (ii) $\bar{\mathcal{R}}_n \notin$ over-segmentation
  - (iii) $\bar{\mathcal{R}}_n \notin$ under-segmentation
- $NE$ (noise): $[\mathcal{R}_m]$ iff
  - (i) $\mathcal{R}_m \notin$ correct detection
  - (ii) $\mathcal{R}_m \notin$ over-segmentation
  - (iii) $\mathcal{R}_m \notin$ under-segmentation

Pixel-wise weighted average criteria

Let us denote

$$n_{i,\bullet} = \sum_{j=1}^{K} n_{i,j} \qquad \text{and} \qquad n_{\bullet,i} = \sum_{j=1}^{K} n_{j,i} \tag{15}$$

where $K$ is a number of classes (or regions), $n$ is the number of pixels in the test set, $n_{i,j}$ is the number of pixels interpreted as the $i$-th class but belonging to the $j$-th class. $\hat{\imath}$ is either $i$ for supervised tests or mapping of the $i$-th class ground truth into an interpretation segment based on the Munkres assignment algorithm [21] for unsupervised tests. The following pixel-wise criteria are implemented:

- $O$ (omission error, the overall ratio of wrongly interpreted pixels):

$$O = \frac{1}{n}\sum_{i=1}^{K} O_i = \frac{1}{n}\sum_{i=1}^{K}(n_{\bullet,i} - n_{\hat{\imath},i}) \quad \in [0,1], \tag{16}$$

  where $O_i$ is the $i$-th class omission error.

- $C$ (commission error, the overall ratio of wrongly assigned pixels):

$$C = \frac{1}{n}\sum_{i=1}^{K} C_i = \frac{1}{n}\sum_{i=1}^{K}(n_{\hat{\imath},\bullet} - n_{\hat{\imath},i}) \quad \in [0,1], \tag{17}$$

  where $C_i$ is the $i$-th class commission error.

- $CA$ (the weighted average class accuracy):

$$CA = \frac{1}{n}\sum_{i=1}^{K} \frac{n_{\hat{\imath},i} n_{\bullet,i}}{n_{\bullet,i} + n_{\hat{\imath},\bullet} - n_{\hat{\imath},i}} \quad \in [0,1] \tag{18}$$

- $CO$ (recall - the weighted average correct assignment):

$$CO = \frac{1}{n}\sum_{i=1}^{K} n_{\bullet,i} CO_i = \frac{1}{n}\sum_{i=1}^{K} n_{\hat{\imath},i} \quad \in [0,1], \tag{19}$$

- $CC$ (precision, object accuracy, overall accuracy):

$$CC = \frac{1}{n}\sum_{i=1}^{K} n_{\bullet,i} CC_i = \frac{1}{n}\sum_{i=1}^{K} \frac{n_{\hat{\imath},i} n_{\bullet,i}}{n_{\hat{\imath},\bullet}} \quad \in [0,1], \tag{20}$$

- $I$ (type I error):

$$I = \frac{1}{n}\sum_{i=1}^{K}(n_{\bullet,i} - n_{\hat{\imath},i}) = 1 - CO \quad \in [0,1], \tag{21}$$

- $II$ (type II error):

$$II = \frac{1}{n}\sum_{i=1}^{K} \frac{n_{\hat{\imath},\bullet} n_{\bullet,i} - n_{\hat{\imath},i} n_{\bullet,i}}{n - n_{\bullet,i}} \quad \in [0,1], \tag{22}$$

- $EA$ (mean class accuracy estimate):

$$EA = \frac{1}{n}\sum_{i=1}^{K}\frac{2n_{\hat{i},i}n_{\bullet,i}}{n_{\bullet,i}+n_{\hat{i},\bullet}} \quad \in [0,1], \tag{23}$$

- $MS$ (mapping score, emphasizes the error of not recognizing the test data):

$$MS = \frac{1}{n}\sum_{i=1}^{K}(1.5n_{\hat{i},i} - 0.5n_{\hat{i},\bullet}) \quad \in [-0.5,1], \tag{24}$$

- $RM$ (root mean square proportion estimation error):

$$RM = \sqrt{\frac{1}{K}\sum_{j=1}^{K}\left(\frac{n_{j,\bullet}-n_{\bullet,j}}{n}\right)^2} = \frac{1}{n}\sqrt{\frac{1}{K}\sum_{j=1}^{K}(C_j-O_j)^2} \quad \in [0,\infty), \tag{25}$$

  indicates unbalance between the omission $O_i$ and commission $C_i$ errors, respectively.

- $CI$ (comparison index, includes both types of errors):

$$CI = \frac{1}{n}\sum_{i=1}^{K}n_{\hat{i},i}\sqrt{\frac{n_{\bullet,i}}{n_{\hat{i},\bullet}}} = \frac{1}{n}\sum_{i=1}^{K}n_{\bullet,i}\sqrt{CC_iCO_i} \quad \in [0,1], \tag{26}$$

  where $CC_i, CO_i$ are the object precision and recall. $CI$ reaches its maximum either for the ideal segmentation or for equal commission and commission errors for every region (class).

Consistency error criteria

Let $S_1, S_2$ be two segmentations, $\mathcal{R}_{1,i}$ is the set of pixels corresponding to a region in the $S_1$ segmentation and containing the pixel $i$, $|\mathcal{R}|$ is the set cardinality and $\setminus$ is the set difference. A refinement tolerant measure error was defined [19] at each pixel $i$:

$$\epsilon_i(S_1,S_2) = \frac{|\mathcal{R}_{1,i}\setminus\mathcal{R}_{2,i}|}{|\mathcal{R}_{1,i}|}. \tag{27}$$

This non-symmetric local error measure encodes a measure of refinement in only one direction. Two error measures for the entire image are defined: the Global Consistency Error, $GCE$, forces all local refinements to be in the same direction while the Local Consistency Error, $LCE$, allows refinement in both directions.

- $GCE$ (global consistency error):

$$GCE(S_1,S_2) = \frac{1}{n}\min\left\{\sum_i \epsilon_i(S_1,S_2)\,,\ \sum_i \epsilon_i(S_2,S_1)\right\} \tag{28}$$

- $LCE$ (local consistency error):

$$LCE(S_1, S_2) = \frac{1}{n} \sum_i \min\{\epsilon_i(S_1, S_2)\,,\, \sum_i \epsilon_i(S_2, S_1)\} \tag{29}$$

$$LCE, GCE \in [0, 1], \qquad LCE \leq GCE \tag{30}$$

#### 4.1.2 Comparative segmentation algorithms

Thanks to the Prague benchmark system we benefit of several segmentation results which allow us to make a comparative analysis of our method. The different algorithms which have been run on the same benchmark data sets are listed and briefly described below:

- GMRF (Gauss MRF model) with EM [12]:
  Single decorrelated monospectral texture factors are assumed to be represented by a set of local Gaussian Markov random field (GMRF) models evaluated for each pixel centered image window and for each spectral band. The segmentation algorithm, based on the underlying Gaussian mixture (GM) model, operates in the decorrelated GMRF space of parameters. The algorithm starts with an oversegmented initial estimation which is adaptively modified until the optimal number of homogeneous texture segments is reached.

- AR3D (3-D Auto Regressive model) with EM [13]:
  This algorithm is similar to the previous one, but the GMRF modeling is replaced by a three dimensional auto-regressive model.

- JSEG [8]:
  The method consists of two independent steps, color quantization and spatial segmentation. In the first step, colors in the image are quantized to several representative classes that can be used to differentiate regions in the image. The image pixels are then replaced by their corresponding color class labels, thus forming a class-map of the image. The subsequent spatial segmentation step applies to the class-map, as to obtain an image, namely the "$J$-image", where high and low values correspond to possible boundaries and interiors of color-texture regions. A region growing method is then used to provide the final segmentation on the basis of a multiscale $J$-images.

- Blobworld (a system for segmentation and CBIR) [1, 4]:
  The algorithm which we will refer to as BW is the basic segmentation tool used in the content-based image retrieval system *blobworld* [4]. Each image is segmented into regions by fitting a mixture of Gaussians to the data in a joint color-texture-position feature space by means of an EM algorithm. Each region ("blob") is then associated with color and texture descriptors, where the textural features taken in consideration are contrast, anisotropy and polarity. Finally, the optimal number of Gaussian components is automatically selected by means of the Minimum Description Length (MDL) criterion. Further details about the segmentation algorithm can be found in [1].

- EDISON (Edge Detection and Image SegmentatiON system) [5]:
  This algorithm is based on the fusion of two basic vision operations, that is image segmentation and edge detection, the former based on global evidence, the latter focused on local information. This integration is realized by embedding the discontinuity (edge) information into to the region formation process, and then using again it to control a post-processing region fusion. In particular EDISON combines the *mean shift* based segmentation [6] with a generalization of the traditional Canny edge detection procedure [3] which employs the confidence in the presence of an edge [20]. For more details about the EDISON algorithm see [5].

#### 4.1.3 Texture mosaic segmentation results

Two versions of the proposed segmentation method were tested on the data set, referred to as TFR and TFR+, which are associated with the two definitions of region gain, see Eq.8 and Eq.12 respectively. The choice of the setting parameters for the two implementations was the same. The number of partially-defined (color) states was 24, that is the CBC/TS-MRF performed a 24-class color-based segmentation for all the images. The number was not optimized but just chosen according to the heuristic rule that takes the double of the maximum number of expected textures in the image, as suggested in Section 3.3. Actually for all the images the number of textures is always less than 12, so we assumed this information to be known and fixed the parameter to 24. Indeed, we have run some tests with different numbers of quantization colors and found only slightly different results. The same maximum number of expected textures is also used as $K$ in the subsequent $k$-means clusterings during the SBC step, and similar considerations about reliability hold in this case as well.

The benchmark data set is composed of twenty different $512 \times 512$ texture mosaics, ten of which are shown in Figures 5-14 together with the associated ground-truth and the segmentations performed by the different methods above mentioned. The numerical comparison, according to the benchmark criteria (see Section 4.1.1), is summarized in Tab.1.

The system provides a comparison w.r.t. a large number of indicators, some of which are region-based, some others are pixel-wise accuracy indicators, and a few of them give a measure of consistency. A complete description of all the parameters, as well as all the results presented here, can be found on the system webpage [14].

The interpretation of the numerical results in Tab.1 may appear ambiguous in some regards, since (of course) no algorithm outperforms uniformly all the others. However it can be easily recognized that the two versions of TFR seem to outperform the other ones in most of the cases, with TFR+ being generally better than TFR. Indeed, the visual inspection of the different segmentations shown in Figures 5-14 allows an easier interpretation, showing clearly the superior performances of the TFR algorithms, especially when the textures present very low frequency patterns, which are known to be more difficult to reveal.

The most evident drawback of the reference methods is the tendency to over-segment, as can be deduced by visually inspecting the segmentations, and confirmed also by the over-segmentation indicator $OS$ (see Tab.1). At the opposite, TFR has a slight tendency to under-segment, $US = 23.99$ (see Tab.1), w.r.t. the other ones, while TFR+ has the most balanced behaviour, keeping quite low

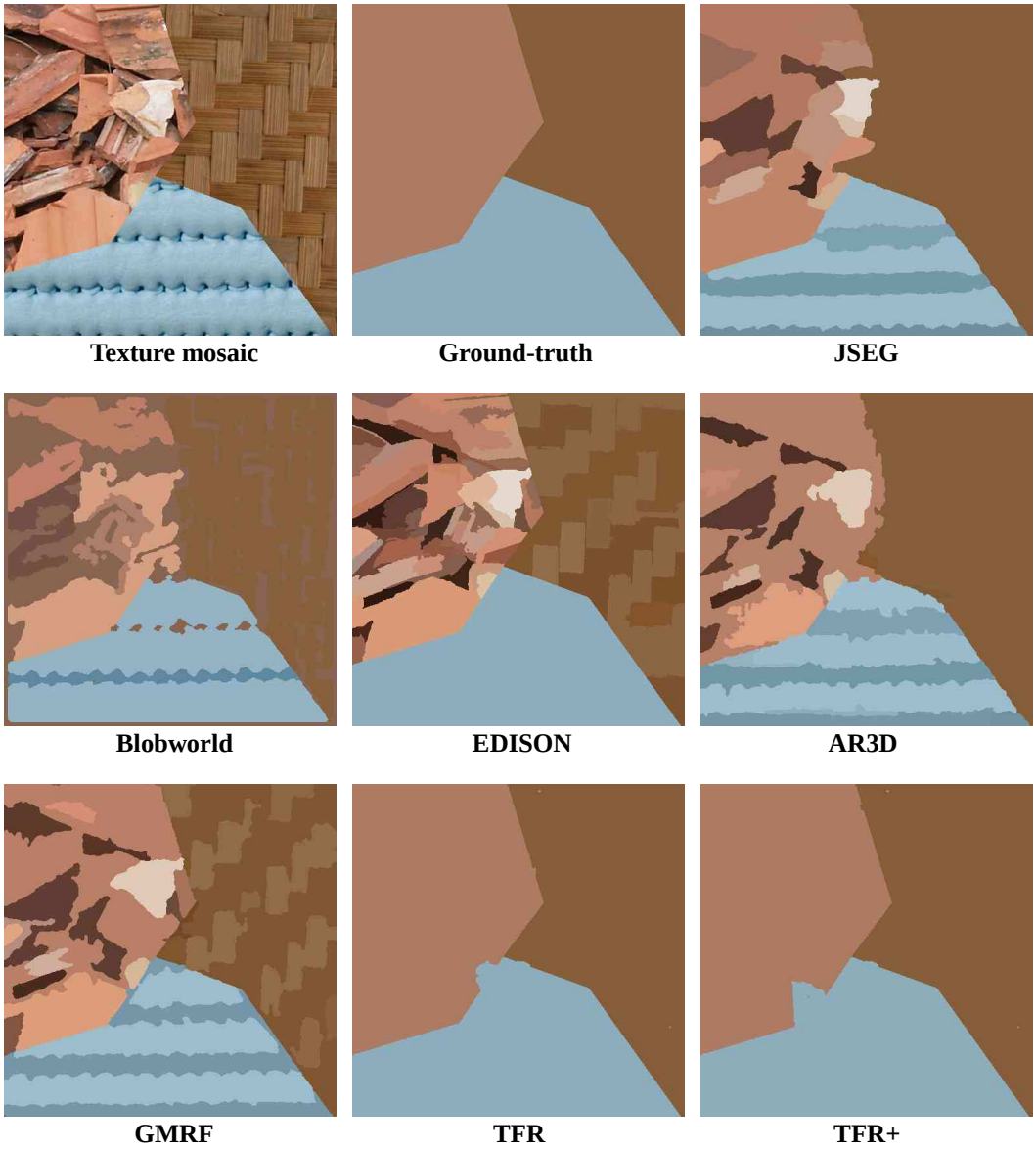



**Figure 5: Texture mosaic No.1: data, ground-truth and several segmentations.**

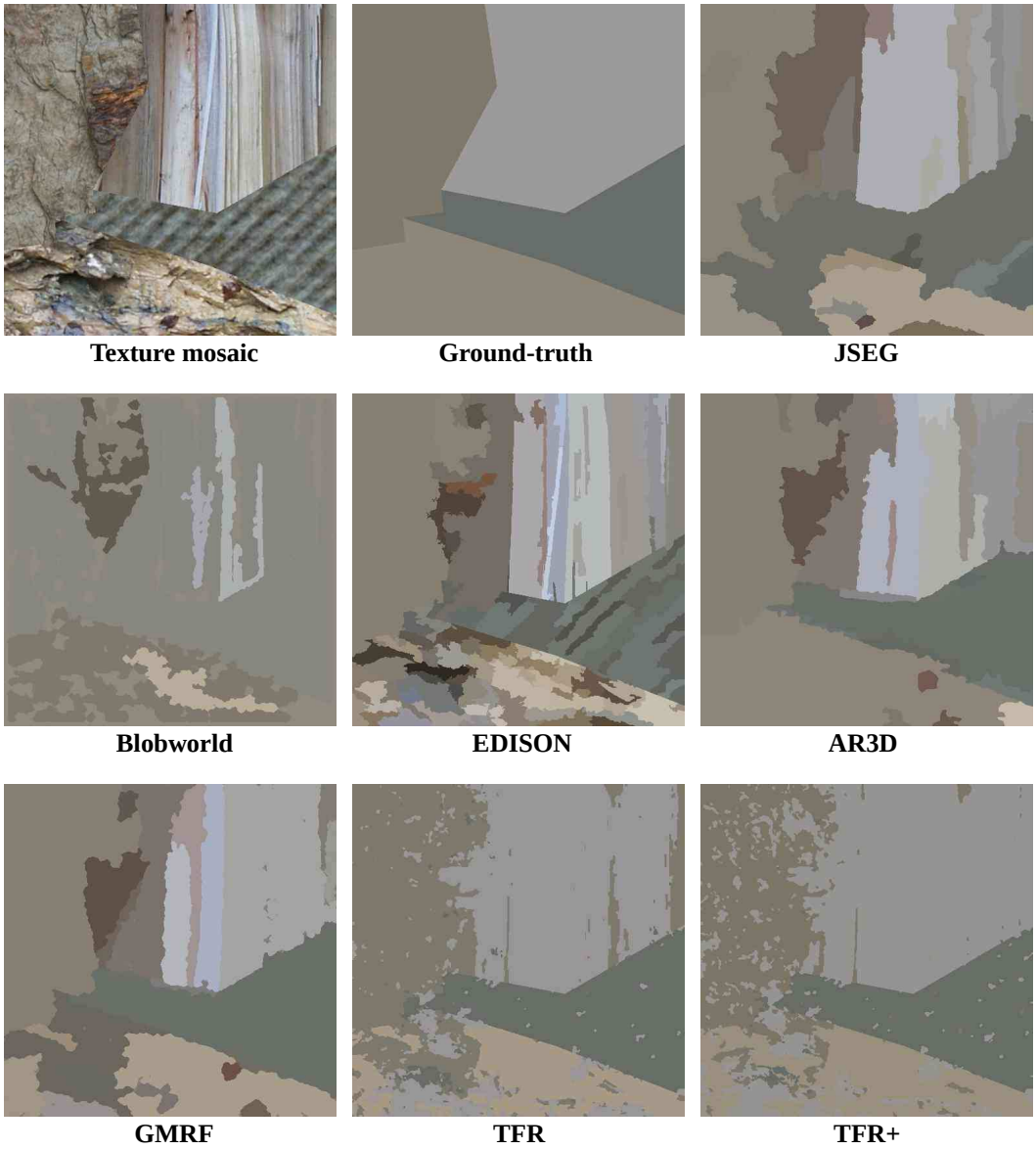



**Figure 6: Texture mosaic No.2: data, ground-truth and several segmentations.**

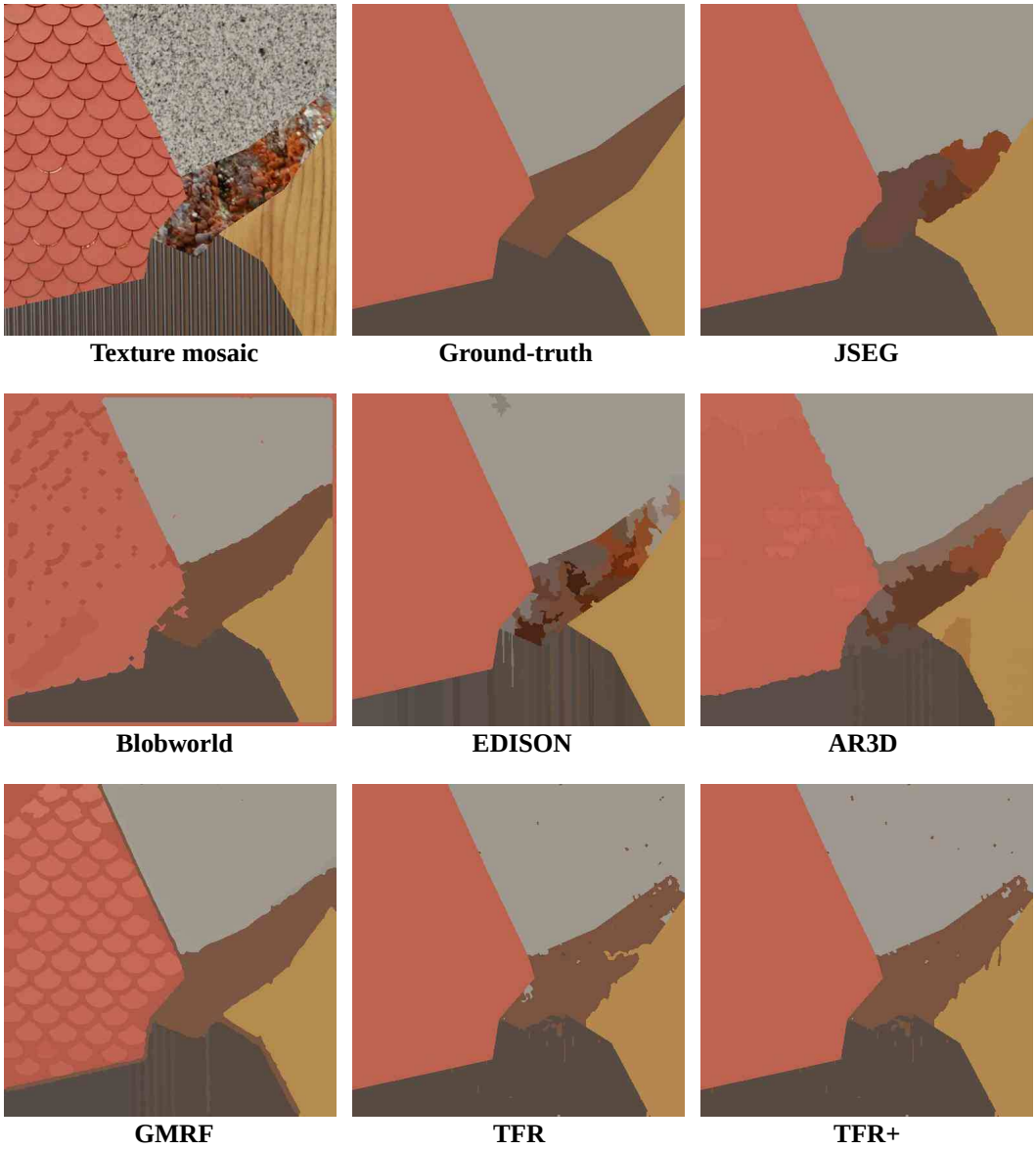



**Figure 7: Texture mosaic No.3: data, ground-truth and several segmentations.**

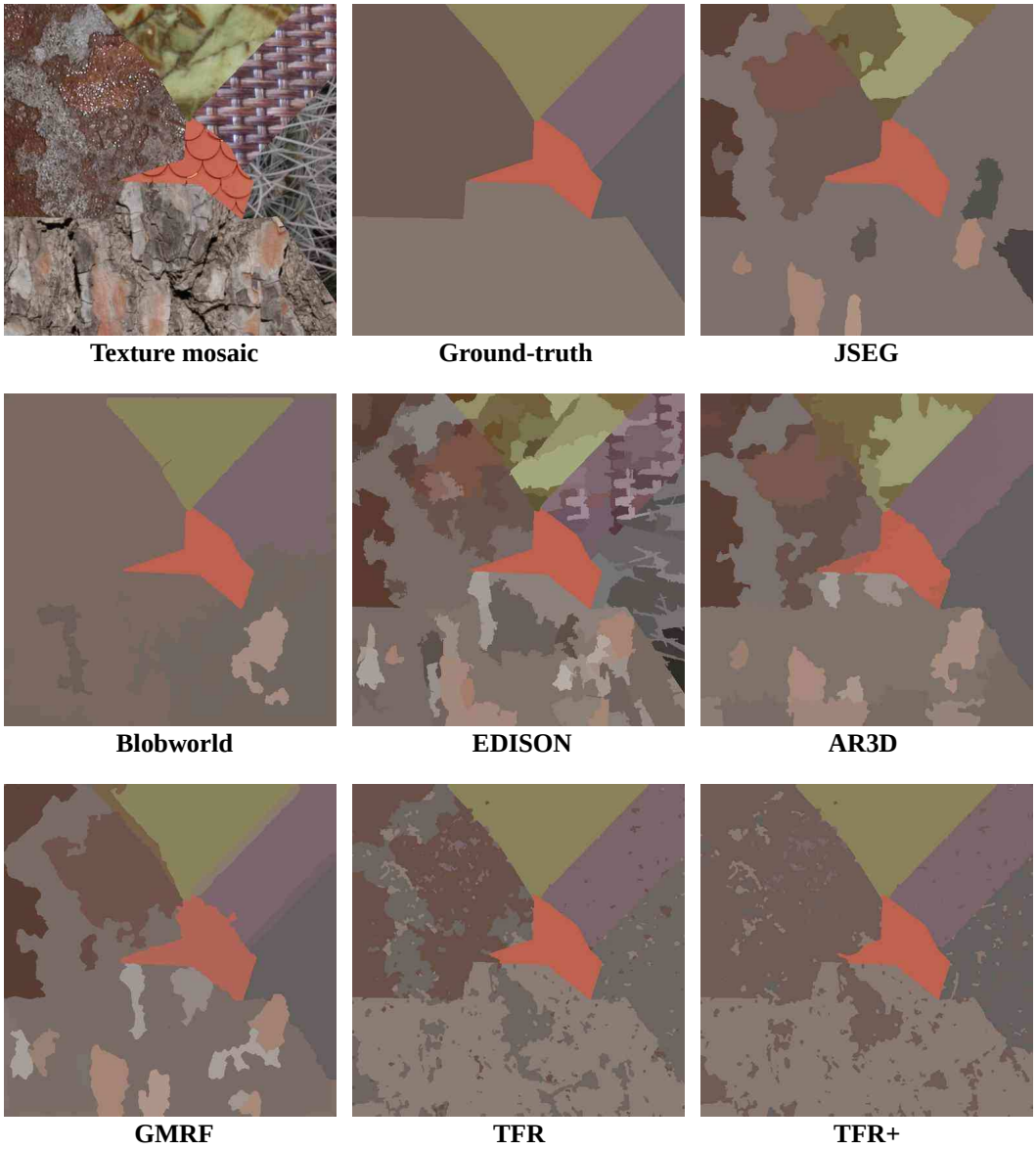



**Figure 8: Texture mosaic No.4: data, ground-truth and several segmentations.**

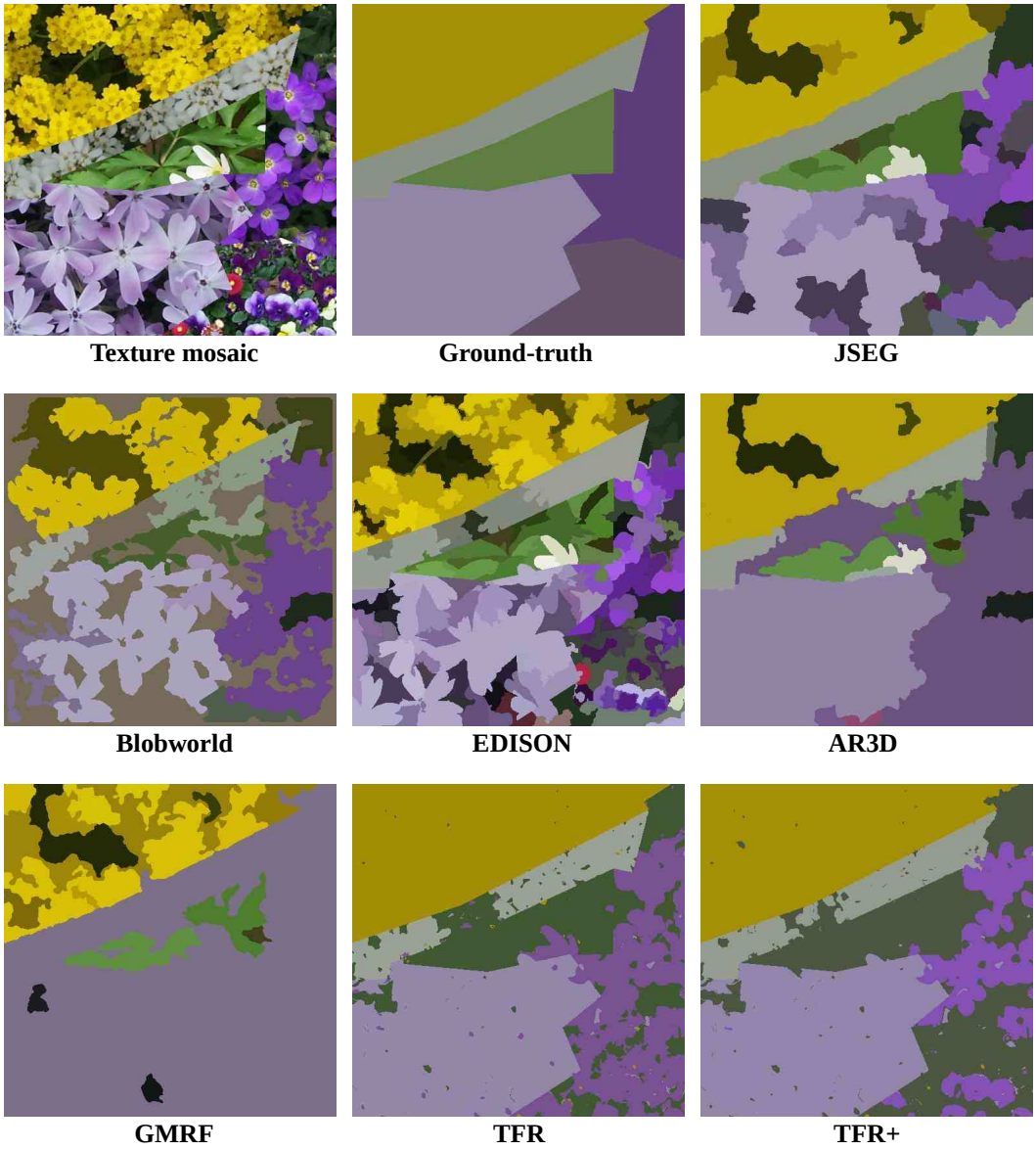



**Figure 9: Texture mosaic No.12: data, ground-truth and several segmentations.**

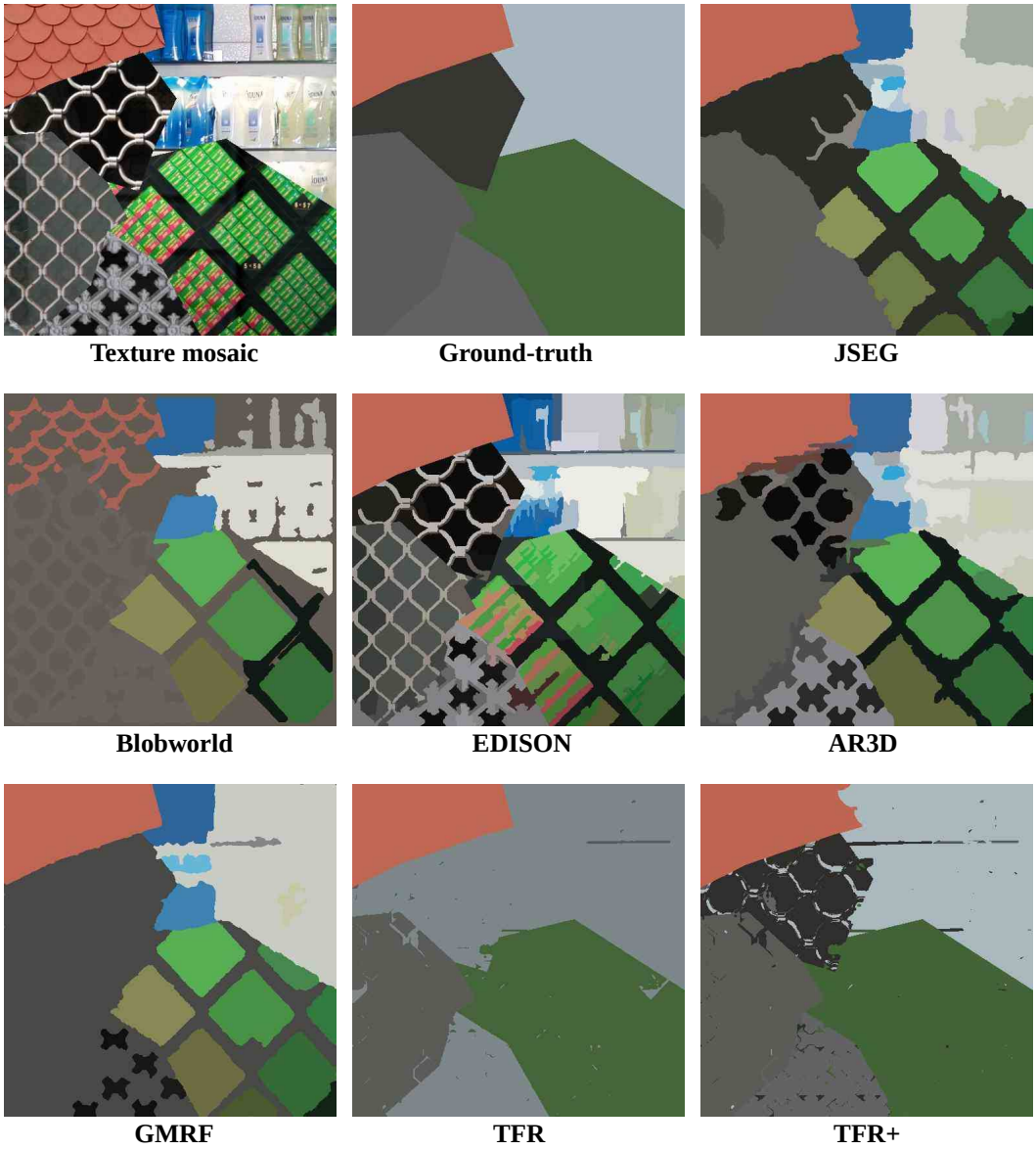



**Figure 10: Texture mosaic No.14: data, ground-truth and several segmentations.**

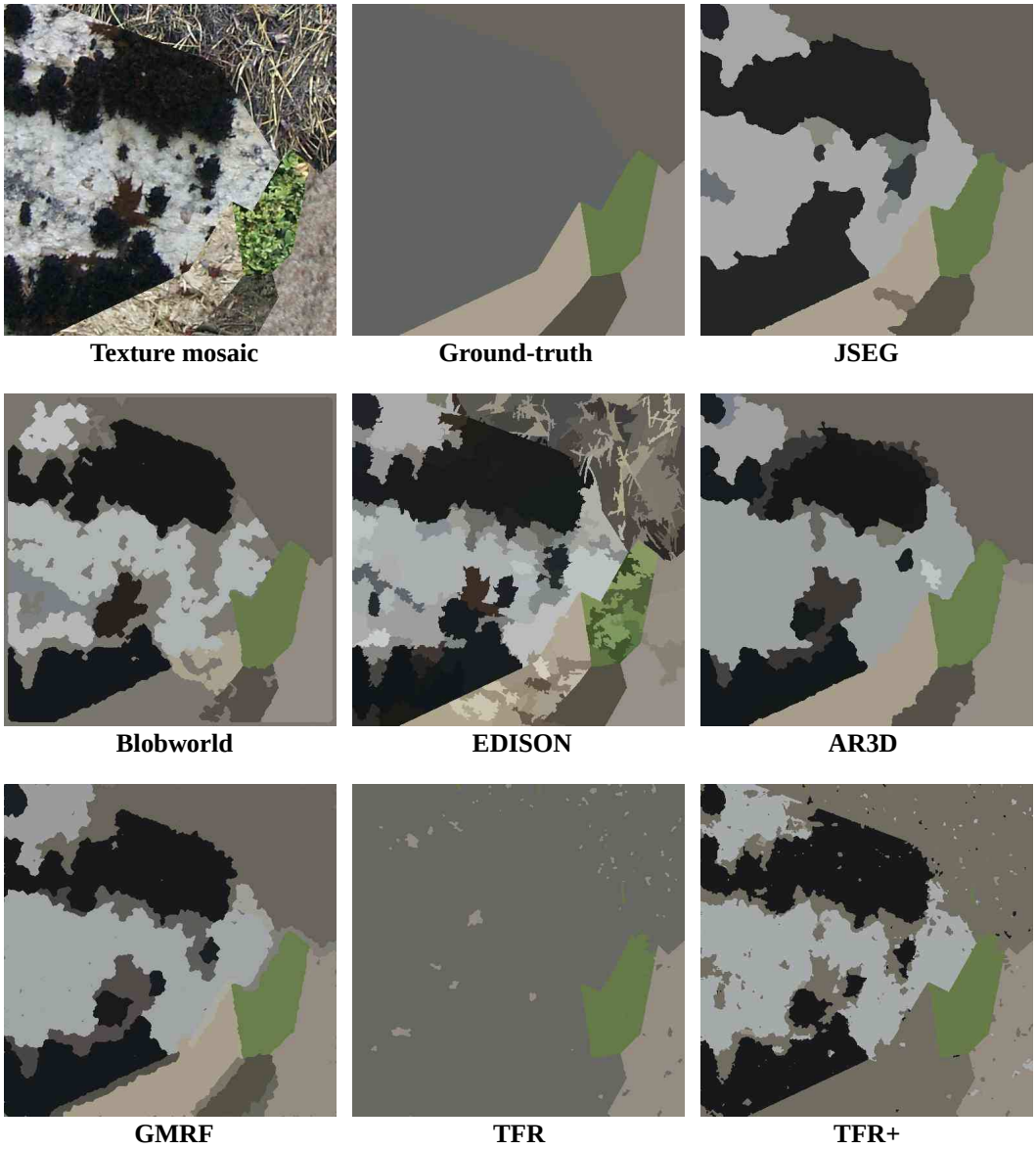



**Figure 11: Texture mosaic No.15: data, ground-truth and several segmentations.**

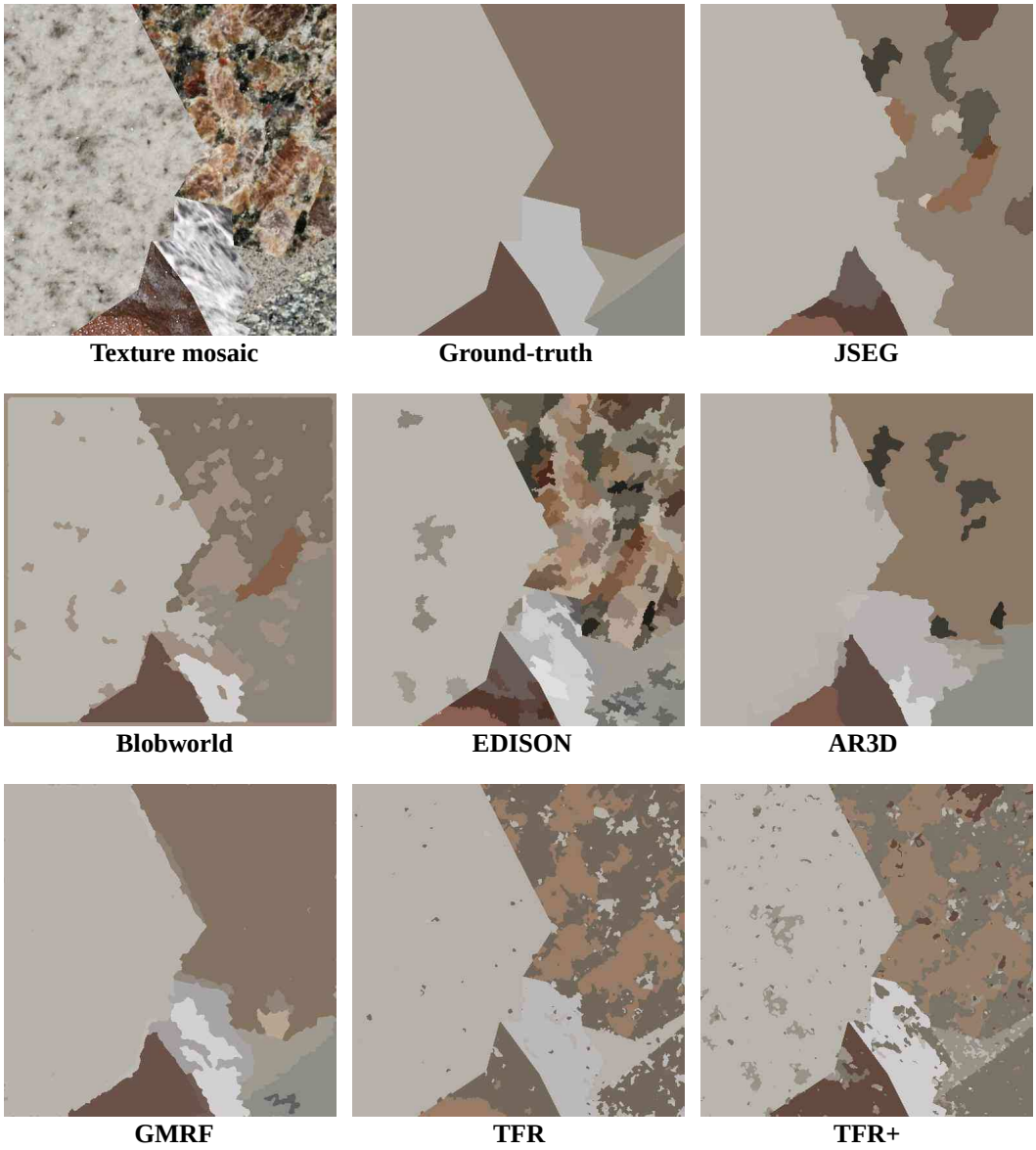



**Figure 12: Texture mosaic No.18: data, ground-truth and several segmentations.**

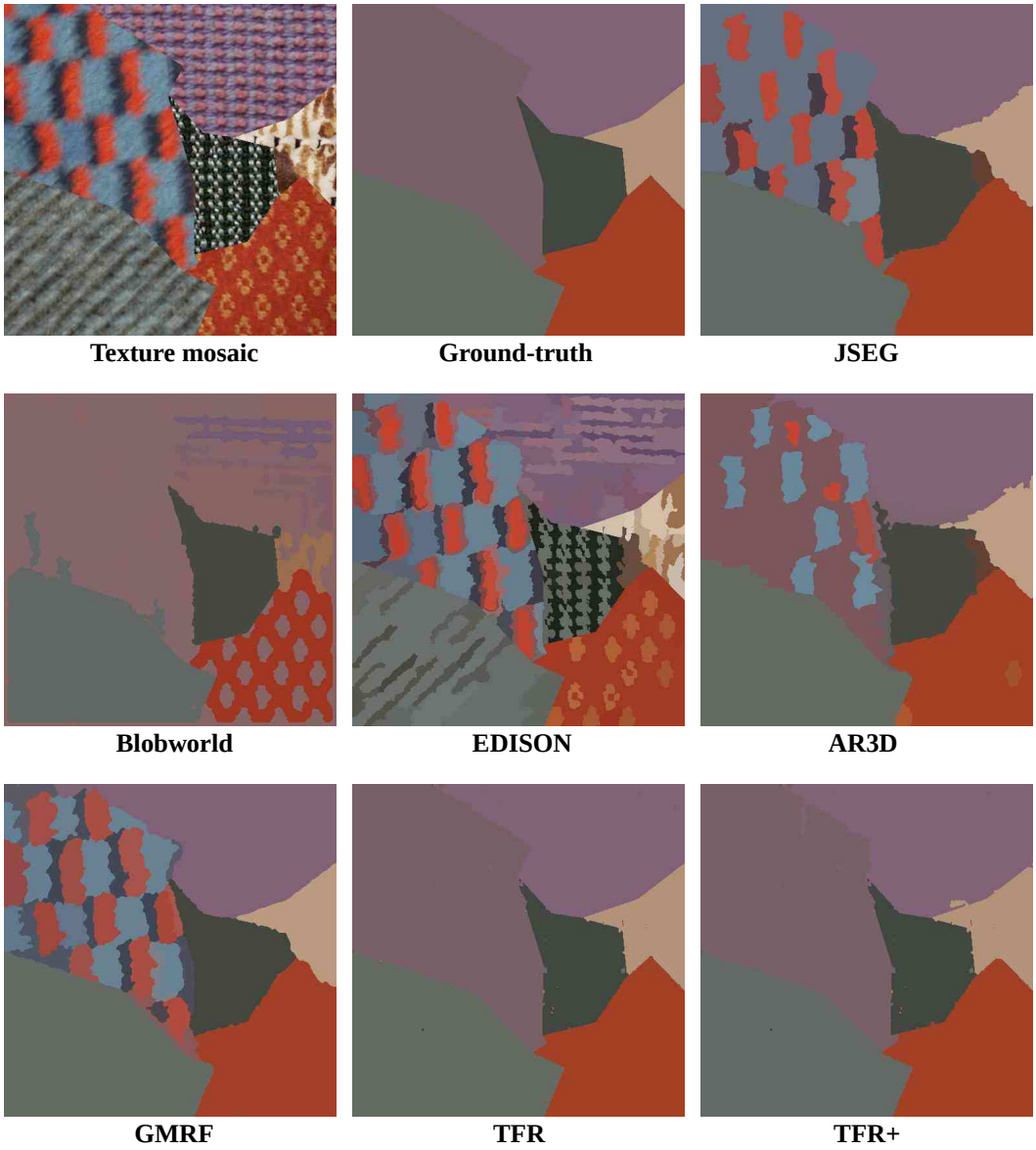



**Figure 13: Texture mosaic No.19: data, ground-truth and several segmentations.**

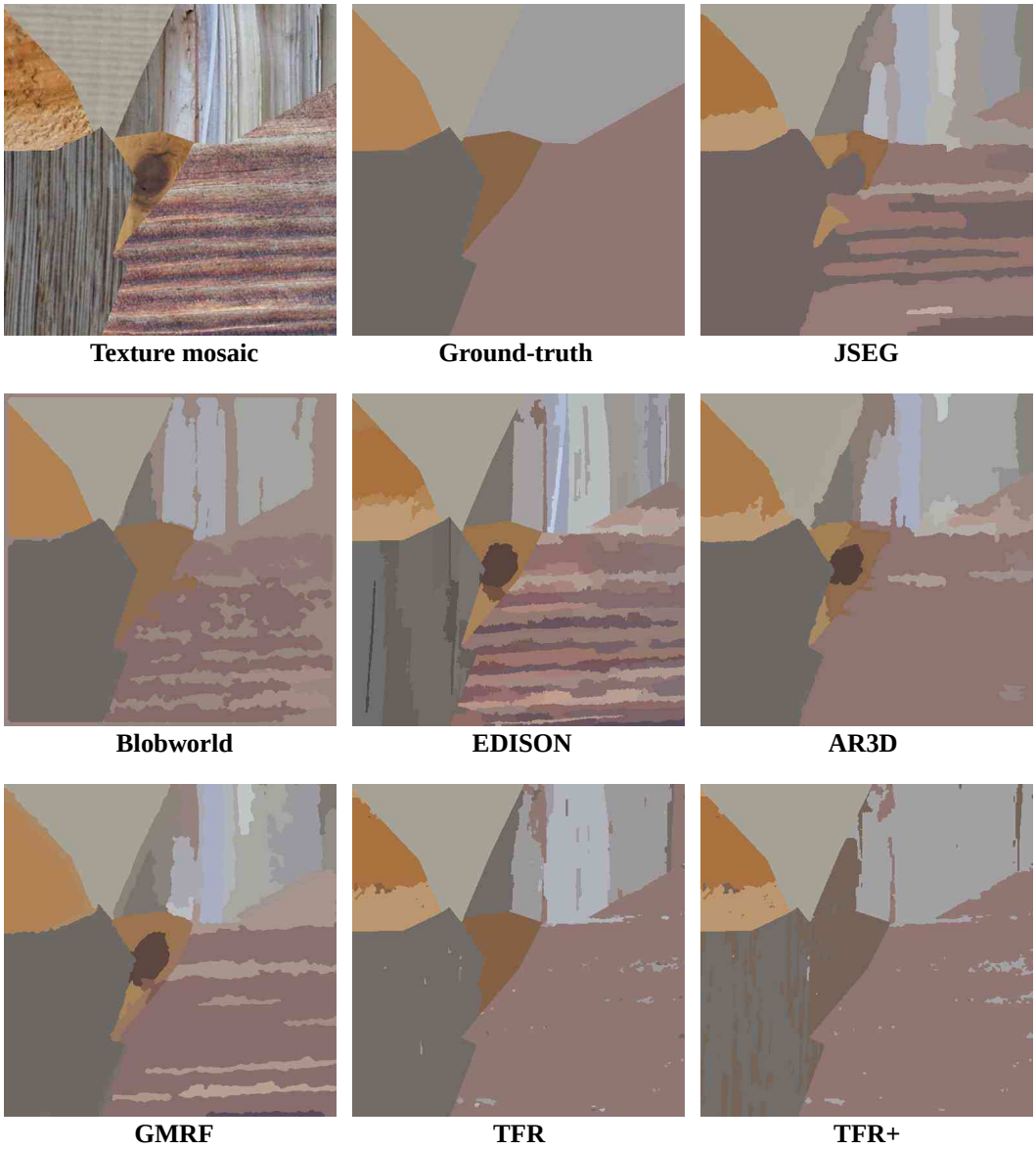



**Figure 14: Texture mosaic No.20: data, ground-truth and several segmentations.**

| | Benchmark – Colour | | | | | | |
|---|---|---|---|---|---|---|---|
| | TFR+ | TFR | AR3D | GMRF | JSEG | Blobworld | EDISON |
| ↑ *CS* | **51.25** | 46.13 | 37.42 | 31.93 | 27.47 | 21.01 | 12.68 |
| ↓ *OS* | 5.84 | **2.37** | 59.53 | 53.27 | 38.62 | 7.33 | 86.91 |
| ↓ *US* | 7.16 | 23.99 | 8.86 | 11.24 | 5.04 | 9.30 | **0.00** |
| ↓ *ME* | 31.64 | 26.70 | 12.54 | 14.97 | 35.00 | 59.55 | **2.48** |
| ↓ *NE* | 31.38 | 25.23 | 13.14 | 16.91 | 35.50 | 61.68 | **4.68** |
| ↓ *O* | **23.60** | 27.00 | 35.19 | 36.49 | 38.19 | 43.96 | 68.45 |
| ↓ *C* | 22.42 | 26.47 | 11.85 | 12.18 | 13.35 | 31.38 | **0.86** |
| ↑ *CA* | **67.45** | 61.32 | 59.46 | 57.91 | 55.29 | 46.23 | 31.19 |
| ↑ *CO* | **76.40** | 73.00 | 64.81 | 63.51 | 61.81 | 56.04 | 31.55 |
| ↑ *CC* | 81.12 | 68.91 | 91.79 | 89.26 | 87.70 | 73.62 | **98.09** |
| ↓ *I.* | **23.60** | 27.00 | 35.19 | 36.49 | 38.19 | 43.96 | 68.45 |
| ↓ *II.* | 4.09 | 8.56 | 3.39 | 3.14 | 3.66 | 6.72 | **0.24** |
| ↑ *EA* | **75.80** | 68.62 | 69.60 | 68.41 | 66.74 | 58.37 | 41.29 |
| ↑ *MS* | **65.19** | 59.76 | 58.89 | 57.42 | 55.14 | 40.36 | 31.13 |
| ↓ *RM* | 6.87 | 7.57 | 4.66 | 4.56 | 4.62 | 7.52 | **3.09** |
| ↑ *CI* | **77.21** | 69.73 | 73.15 | 71.80 | 70.27 | 61.31 | 50.29 |
| ↓ *GCE* | 20.35 | 15.52 | 12.13 | 16.03 | 18.45 | 31.16 | **3.55** |
| ↓ *LCE* | 14.36 | 12.03 | 6.69 | 7.31 | 11.64 | 23.19 | **3.44** |

**Table 1: Prague texture segmentation benchmark results. Up arrows indicate that larger values of the parameters are better; down arrows, the opposite. Bold numbers indicate the best one depending on the criterion.**

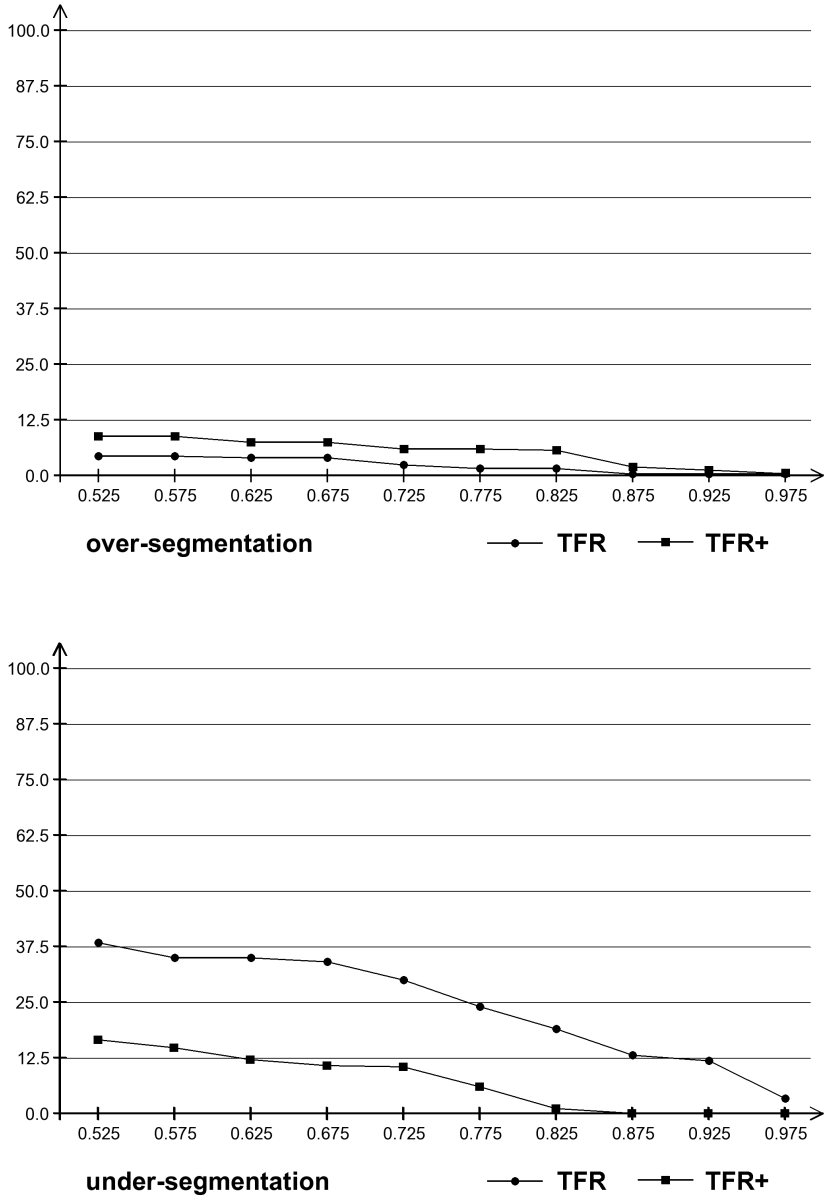



**Figure 15: Over- and under-segmentation w.r.t. the parameter $k \in [0.5, 1]$, for TFR and TFR+.**

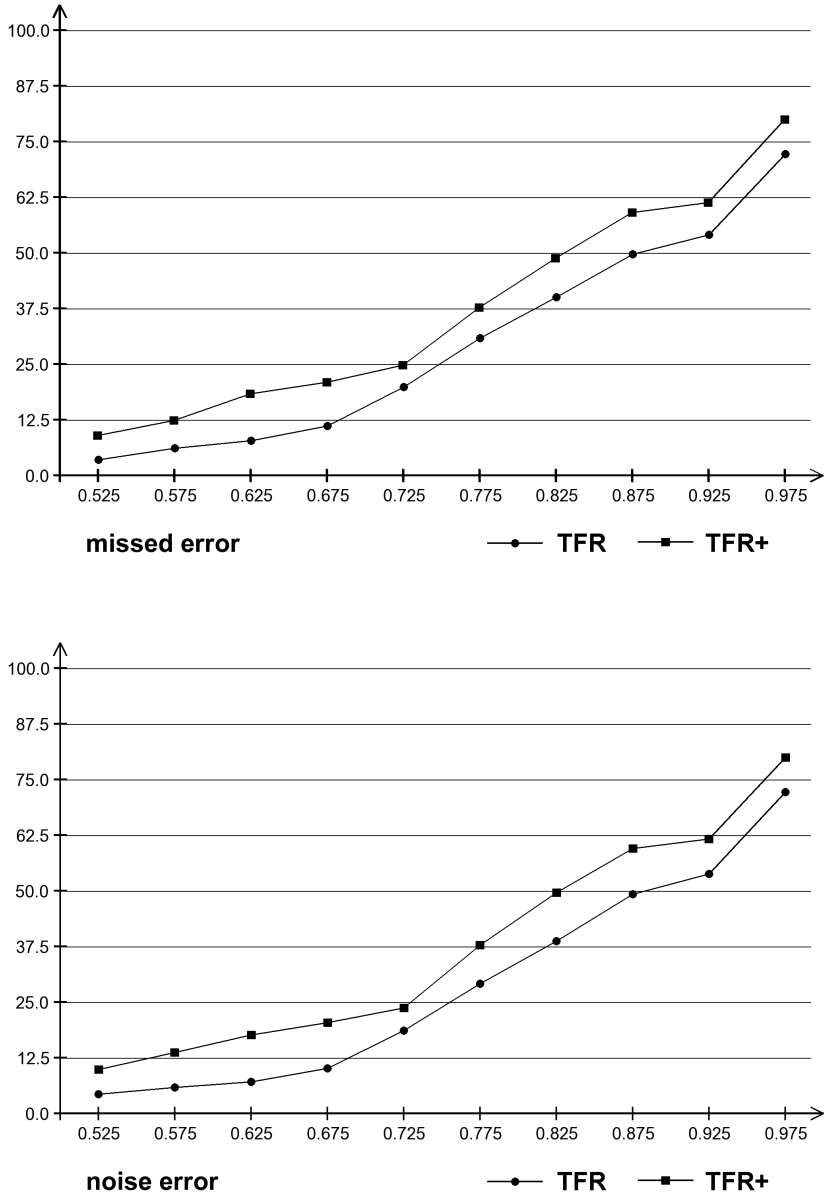



**Figure 16: Missed and noise w.r.t. the parameter $k \in [0.5, 1]$, for TFR and TFR+.**

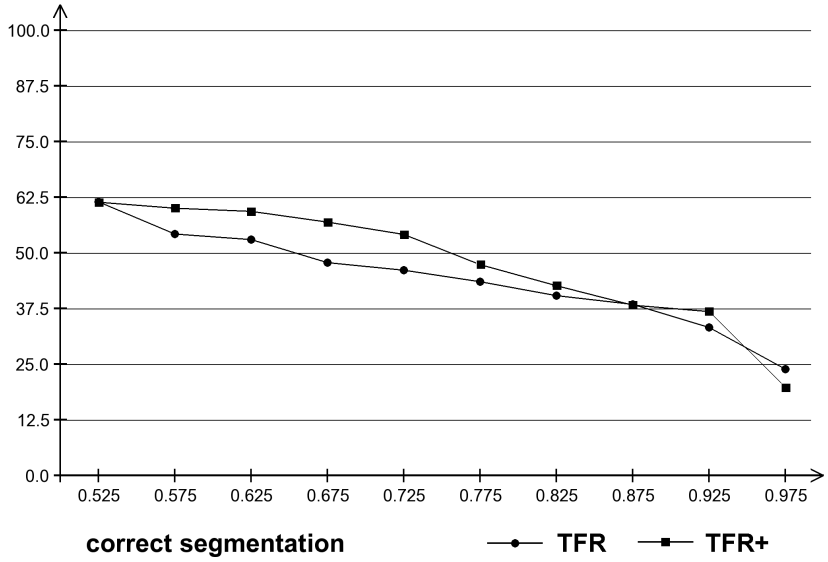



**Figure 17: correct detection w.r.t. the parameter $k \in [0.5, 1]$, for TFR and TFR+.**

both $OS$ and $US$. EDISON algorithm always over-segments, meaning that it is not able at all to model the macro-textural features, obviously it is the best w.r.t. $US$ but the worst w.r.t. $OS$.

Finally, in Figures 15-17 the behaviour of several accuracy indicators w.r.t. the parameter $k \in [0.5, 1]$ (see Section 4.1.1) is shown for both TFR and TFR+.

## 4.2 Application to remote-sensing images covering forest areas

In this section an application of the proposed method on remote-sensing data is presented. It is the case of high resolution (50cm) aerial images covering forest areas, which match well with the proposed modeling since they present different relevant texture patterns with acceptable stationarity. Such images are curtesy of the "French Forest Inventory" (IFN).

We present two experiments. The former, see Fig.18, refers to an area composed of several classes of trees plus no tree lands and shadows. Since we have no ground-truth related to these data, we build up the latter experiment where a mosaic image was obtained which is composed of four square subimages, see Fig.19. Three of them represent different quasi stationary tree textures, while the last one (bottom-left) is a mixing of a urban class and one of the other (bottom-right) tree textures.

We experimented only the case of TFR+, since it has been shown to be better than TFR in the previous section. Also no comparative algorithms have yet been tested on these data, and eventually we can only make conjectures about the performances of TFR+. A comparison with another method currently under development could be made later.

The $1024 \times 1024$ forest image and the associated 5-class TFR+'s segmentation are shown in Fig.18. One class represents just the shadows, one is associated with low vegetation areas, the remaining three correspond to different tree patterns. The segmentation seems to be quite promising according to a visual inspection. Indeed, in order to obtain such good result, a slight modification of the TFR+ algorithm was necessary. In fact, the proposed optimization schemes (meaning both TFR and TFR+) are sensitive to the presence of continuous regions, like background colors, because these are typically large and, hence, work as collectors of other regions. This becomes a critical problem when different textures have the same background color and share a long contour, where we can found many of such regions which cross the border and, therefore, link the textures forcing a merging. Unfortunately this was the case of the shadow regions present in the image. For this reason we decided to simply detect the background regions (just the shadows, in this case) after the CBC step, and ignore them from the subsequent steps (SBC and region merging).

Instead, in the latter experiment such modification was not necessary. The results are encouraging in this case as well. In particular, from the segmentation shown in Fig.19, we can see that the three different tree patterns have been detected with satisfactory precision. As for the mixed urban-trees area (bottom-left), the urban elements are assigned with a fourth class, while the trees are largely assigned with the correct tree class (that at bottom-right).

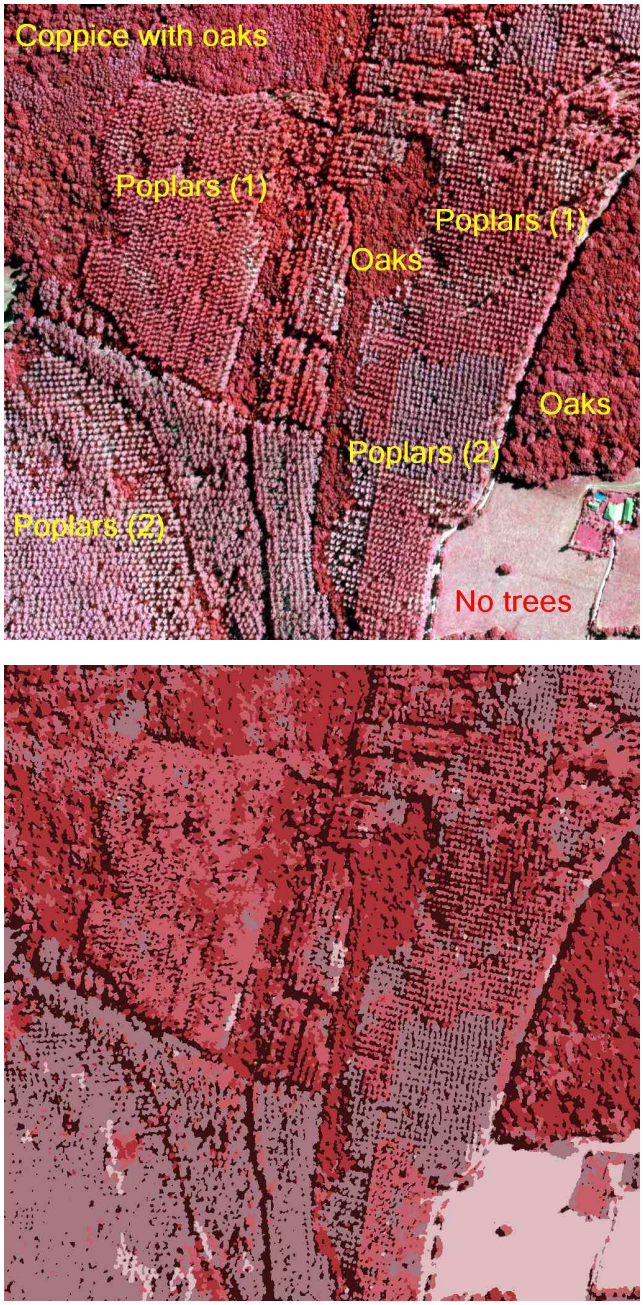



**Figure 18: Top: Forest image, south of Burgundy, France. ©IFN. Bottom: Segmentation obtained by the TFR+ algorithm (5 classes: two kinds of poplars, oaks, no trees, and shadows).**

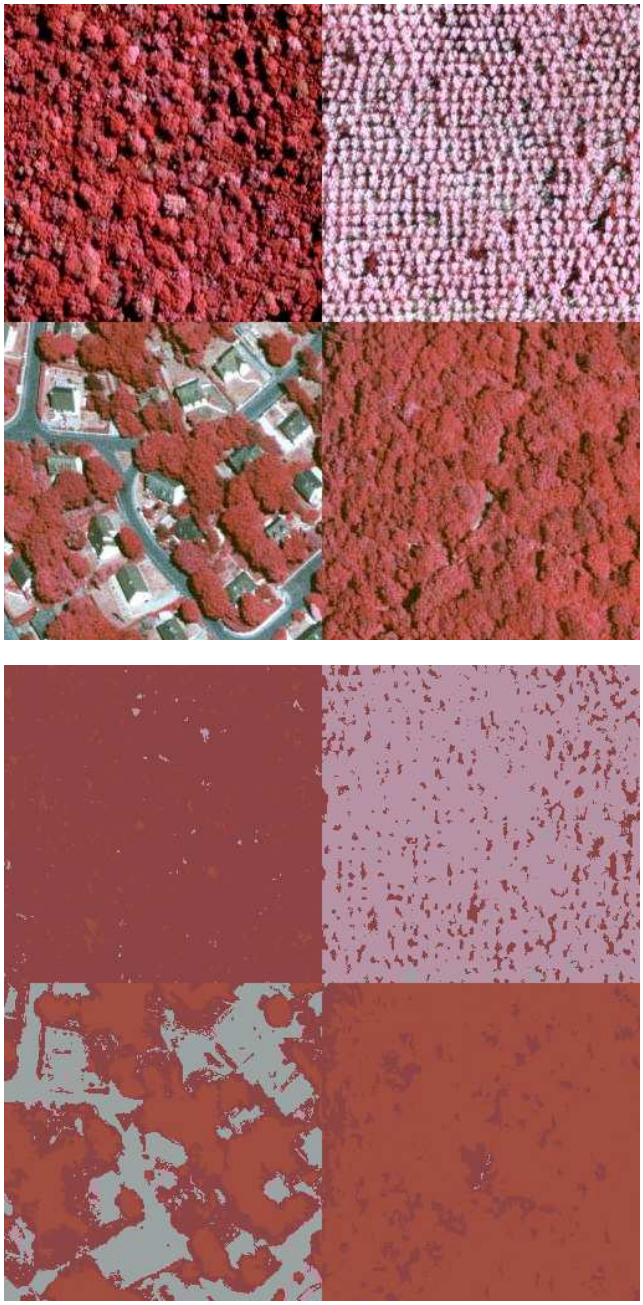


**Figure 19: Top: Mosaic of different kinds of remotely sensed forest patterns, south of Burgundy ©IFN. Bottom: Segmentation obtained by TFR+ (4 classes).**

# 5 Conclusion

In this research report we have presented a hierarchical finite-state model for texture representation which is particularly suited for unsupervised segmentation, as shown by the results the two proposed optimization schemes, TFR and TFR+. Since the model is region-based and discrete, the first step of the TFR is a color-based segmentation, realized by TS-MRF, that provides the rough discrete approximation of the original data to be fitted with the texture model at region level. The fitting is then performed in two sequential steps, the former (SBC) which focuses on defining individually the states of the model, the latter aimed at relating them hierarchically according to the scale of the corresponding regions and their mutual spatial interaction. In particular, in order to control the bottom-up building of the model structure, two different *region gain* parameters have been defined.

Performance assessment of the proposed segmentation algorithms was achieved experimenting with the texture mosaic data sets provided by the *Prague Texture Segmentation Datagenerator Benchmark* [14] that scores the several algorithms which make use of its data sets w.r.t. several accuracy indicators.

From both numerical evidence and visual inspection of the segmentation results, it appears that the two proposed versions of TFR outperform all the comparative algorithms. In our opinion, the better performances are basically due to the fact that most of the textures considered in the experiments contain spatial correlations at multiple scales and, therefore, can only be captured by means of a multiscale model and possibly working at region level. All the methods using a pixel-based texture modeling present serious limitations especially when they have to represent macro-textural features, which is the case of the most of the texture models that can be found in the current literature.

The experimental results also show that the latter proposed TFR version, i.e. the TFR+, which makes use of a region gain that generalizes the other one by including a Kullback-Leibler divergence between the spatial distribution of the regions, clearly outperforms the former. In particular, the main difference is the reduction of the under-segmentation phenomenon observed for TFR.

Eventually, the main advantages of the proposed solutions (both TFR and TFR+) can be summarized as follows.

- **Scalable**. The region-hierarchical underlying model allows to provide the user with a nested hierarchical segmentation where each single segmentation corresponds to a given scale whose selection can be left to the user. From a segmentation point of view, this means also that the cluster validation problem is only partially addressed consistently with a multiscale interpretation of the image.

- **Robust**. Contrary to pixel-based models, due to its region-based formulation, the proposed model is able to represent spatial interactions of variable range with the same complexity order, since they propagate on a region-by-region step rather than on a pixel-by-pixel one. As a consequence the model does not require the specification of any window in order to focus on the spatial interaction range of interest and, eventually, the resulting algorithm is rather robust.

- **Quick**. Another consequence of modeling the image at a region level is the strong reduction of computational load, since the image processing involves regions, instead of pixels, whose number is much lower.

- **Blind**. The algorithm can be considered completely unsupervised even if a few tuning parameters can be chosen properly if some information is known. In particular, the number of segments given by the CBC step, controlling the degree of over-segmentation at this level, can be related to the "expected" number of classes. The number $K$ of clusters of the $k$-means algorithm (at SBC level) can be derived from the same information as well.

On the other hand a few drawbacks of the technique have to be mentioned as well. In particular the discrimination of micro-textural features whose spatial range approaches the pixel size presents the same limitations since the region-wise characterization of the image elements becomes unreliable due to the small size of the regions. Another critical aspect is about the optimization, that is the estimation of the model states with corresponding relationships. In fact, the simultaneous interaction of different Markov processes (each one associated with a spatial direction), make very difficult the propagation of the inter-region interactions during the model fitting of the data. That is why we have proposed the simple TFR schedule for preforming the model optimization.

Finally, apart from the intrinsic limitations discussed above, several other weak points of the algorithm due to its simple implementation may certainly be improved, and the most important ones are briefly presented here.

One point is about the modeling of the spatial distribution of the regions in the evaluation of the Kullback-Leibler divergence in the region gain. We used a simple normal distribution to model regions which, indeed, are typically collections of disjoint subregions. Therefore a mixture model seems to be more suited to this purpose than a simple normal distribution.

Another simple solution which could be replaced by a more effective one is the $k$-means algorithm that is used at the SBC layer. In particular, an important peculiar aspect of the needed clustering should be taken into account: in fact, at any split of a color-state, it is not strictly required that each region is assigned with a state. Indeed, what is more important is to form agglomerates of regions (defining a state) which are quite homogeneous w.r.t. the spatial features, while single regions which do not fit well with any group may be considered alone. In the current implementation we simply fix a number of clusters to be singled out with the $k$-means and, for sure, this could be further improved with an *ad hoc* clustering.

The color-based clustering may be improved as well, by the use of a different split gain to control the TS-MRF tree growth, such that at the end of the fragmentation step the sizes of the elementary connected regions are closer, and this would make more uniform the spatial resolution of the final segmentation.

Another critical problem is about the processing of "continuous" connected regions that appear typically for textures containing background constant-colors. An example is the experiment with the forest images where the shadow regions were quite continuous. We have been forced to make a slight modification to the TFR algorithm for this case. To be more precise, continuous regions are undesirable for two reasons. First, since they are large they occur typically rarely in a texture, sometimes just once, and then a robust ensemble characterization cannot be achieved. Second, when two neighbouring textures have a common color-state which presents such continuous elements, due to their large scale they serve mostly as collectors during the region merging, attracting regions from the two different textures which, eventually, result merged together. For the forest images, we have

detected the shadow regions (they have the lowest spectral response), and then simply excluded them from bottom-up reconstruction step.

Last but not least concern is about the capability of the TPMs to characterize the regions in terms of shape and spatial context. In fact, it is easy to see that the TPMs give a good description of regions which have a linear shape, no matter how polarized in the space. As the shape becomes more complex, the TPM characterization becomes more and more approximated. A possible solution to this problem would be the insertion of a shape-based region decomposition between the CBC and SBC layers.

## Acknowledgments

This work was carried out during the tenure of an ERCIM fellowship (Scarpa's postdoc), and supported by EU MUSCLE project (FP6-507752). The authors would also like to thank the "French Forest Inventory" (IFN) for providing the remote-sensing data covering the forest areas.

## References

[1] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proceedings of the Sixth International Conference on Computer Vision*, pages 675–682, Bombay, India, January 1998.

[2] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, series B*, 48:259–302, 1986.

[3] J. Canny. A computational approach to edge detection. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 8:679–698, 1986.

[4] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Third International Conference on Visual Information Systems*, pages 509–516, Amsterdam, The Netherlands, 1999. Springer.

[5] C. M. Christoudias, B. Georgescu, and P. Meer. Synergism in low level vision. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th International Conference on Pattern Recognition*, volume 4, pages 150–155, Los Alamitos, August 2002.

[6] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 24(5):603–619, May 2002.

[7] C. D'Elia, G. Poggi, and G. Scarpa. A tree-structured Markov random field model for Bayesian image segmentation. *IEEE Transaction on Image Processing*, 12(10):1259–1273, October 2003.

[8] Y. Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 23(8):800–810, 2001.

[9] B. Fischer, C. J. Thies, M. O. Guld, and T. M. Lehmann. Content-based image retrieval by matching hierarchical attributed region adjacency graphs. In *Proc. SPIE*, volume 5370, pages 598–606, San Diego, CA, USA, 2004.

[10] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 6(6):721–741, November 1984.

[11] A. Gersho and R. Gray. *Vector quantization and signal compression*. Kluwer, Boston, MA, 1992.

[12] M. Haindl and S. Mikeš. Model-based texture segmentation. In A. Campilho and M. Kamel, editors, *Image Analysis and Recognition*, Lecture Notes in Computer Science 3212, pages 306–313, Porto, Portugal, 2004. Springer Berlin / Heidelberg.

[13] M. Haindl and S. Mikeš. Colour texture segmentation using modelling approach. In *Proc. 3th ICARP*, Lecture Notes in Computer Science 3687, pages 484–491, Bath, UK, 2005.

[14] M. Haindl and S. Mikeš. Prague texture segmentation datagenerator benchmark, 2005. `http://mosaic.utia.cas.cz`.

[15] S. Hese. Segmentation of forest stands in very high resolution stereo data. In *Proc. IEEE International Geoscience and Remote Sensing Symposium*, volume 4, pages 1654–1656, Sidney (AUS), July 2001.

[16] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. W. Bowyer, D. W. Eggert, A. W. Fitzgibbon, and R. B. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 18(7):673–689, 1996.

[17] Z. Kato, T. C. Pong, and J. C. M. Lee. Color image segmentation and parameter estimation in a Markovian framework. *Pattern Recognition Letters*, 22(3-4):309–321, March 2001.
[18] S. Kullback. *Information theory and statistics*. Dover Publications, New York, 1968.
[19] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings 8th International Conference on Computer Vision*, volume 2, pages 416–423, Vancouver, Canada, July 2001.
[20] P. Meer and B. Georgescu. Edge detection with embedded confidence. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 23(12):1351–1365, 2001.
[21] J. Munkres. Algorithms for assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, March 1957.
[22] W. D. Penny. Kullback-Leibler divergences of normal, gamma, Dirichlet and Wishart densities. Technical report, Wellcome Department of Imaging Neuroscience, University College Longon, 2001.
[23] G. Poggi, G. Scarpa, and J. Zerubia. Supervised segmentation of remote-sensing images based on a tree-structured MRF model. *IEEE Transaction on Geoscience and Remote Sensing*, 43(8):1901–1911, August 2005.
[24] P. Salembier and L. Garrido. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE Transactions on Image Processing*, pages 561–576, April 2000.



Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399