Computational Geosciences
https://doi.org/10.1007/s10596-019-09901-z

ORIGINAL PAPER

Quantitative evaluation of multiple-point simulations using image segmentation and texture descriptors

Mohammad Javad Abdollahifard[1] · Grégoire Mariéthoz[2] · Maryam Ghavim[1]

Received: 9 April 2019 / Accepted: 13 September 2019

Abstract
Continuous growth of multiple-point simulation algorithms for modeling environmental variables necessitates a straightforward, reliable, robust, and distinctive method for evaluating the quality of output images. A good simulation method should produce realizations consistent with the training image (TI). Moreover, it should be capable of producing diverse realizations to effectively model the variability of real fields. In this paper, the pattern innovation capability is evaluated by estimating the coherence map using keypoint detection and matching, without assuming any access to the simulation process. Local binary patterns, as distinctive and effective texture descriptors, are also employed to evaluate the consistency of realizations with the TI. Our proposed method provides absolute measures in the interval [0,1], allowing MPS algorithms to be evaluated on their own. Experiments show that the produced scores are consistent with human perception and robust for different realizations obtained using the same method, allowing for a reliable judgment using a few realizations. While a human observer is highly sensitive to discontinuities and insensitive to verbatim copies, the proposed method considers both factors simultaneously.

Keywords Geostatistics · Variability · Texture analysis · Local binary patterns · Coherence map

1 Introduction

In recent years, multiple-point statistics (MPS) methods have been heavily applied to a wide range of environmental modeling problems including hydrology, hydrogeology, ecology, epidemiology, remote sensing, natural resource estimation, petroleum engineering, and mining [1–3]. The basic concepts of MPS were first introduced by Guardiano and Srivastava [4]. MPS methods are founded on the idea that the features of environmental variables can be modeled using a training image (TI). The simulation grid (SG), which at first contains a limited number of conditioning observations, needs to be completed in a way consistent with both hard data and the features of the TI.

In an attempt to reach a satisfactory compromise between computational burden, pattern reproduction, and between-realization variability, many different simulation methods were proposed in the literature. MPS methods usually proceed by scanning the grid points in a specific order. In order to complete the SG in a consistent manner, at each grid point, available data in a limited neighborhood (called a data-event) is extracted and the TI is sought to find a proper match. Then, some new data are transferred from the found match to the SG.

While pixel-based methods transfer only one pixel (voxel) from each found match [5–10], patch-based methods achieve higher speed and better pattern reproduction performance by transferring a large number of pixels (voxels) at once [11–18]. However, patch-based methods suffer from some important limitations including difficulty of handling conditioning data, low variability, and verbatim copy of the large patches of the TI into the SG (i.e., large portions copied identically from the TI).

In other words, most MPS methods attempt to model the variability of the field by consistently permuting patterns of a limited TI. Although this strategy leads to visually plausible and consistent realizations, it usually underestimates the true variability [19]. The realizations produced using MPS algorithms are usually judged subjectively. While the human

✉ Mohammad Javad Abdollahifard
mj.abdollahi@tafreshu.ac.ir

[1] Electrical Engineering Department, Tafresh University, Tafresh 39518 79611, Iran

[2] Institute of Earth Surface Dynamics, Universite de Lausanne, Lausanne, Switzerland

Published online: 08 November 2019

visual system is highly sensitive to discontinuities, it is insensitive to verbatim copy. For researchers to continue developing and applying new MPS algorithms, there is a need for objective evaluation methods of the simulated fields to allow them for a fair judgment.

Before proceeding into the details of quantitative evaluation methods, a fundamental question should be answered: what are the features of an ideal simulation algorithm? Of course, the features of the produced realizations should be consistent with the TI. In other words, the patterns of the realizations should be similar to the patterns present in the TI. At the same time, the simulation method should be capable of producing diverse realizations to effectively model the variability of real environmental phenomena. A significant drawback associated with MPS methods is that the TI inevitably has limited extent and does not include the full variability of the real field [20]. Simulation method should thus ideally be able to create patterns that are at the same time new and similar to those found in the TI. If a large portion of the TI is repeated in the realization (i.e., verbatim copy), the simulation is insufficient in terms of pattern innovation.

1.1 Background work

A number of quantitative methods for evaluating MPS algorithms have been proposed in the literature. Tan et al. [21] propose a distance function that measures the difference between the multiple-point statistics of two images, I and J. This is done by comparing multiple-point histograms (MPHs) or clustering-based histograms of patterns (CHPs). For a given binary image, the MPH is the table of frequencies of every possible pattern configuration, within a fixed template size [22]. For the example of a binary image with a 4×4 template, the histogram would have $2^{16} = 65{,}536$ bins. If normalized MPH of I and J are denoted by g and h, respectively, then the distance between two images can be computed using the Jensen-Shannon divergence [23]:

$$d(g,h) = \frac{1}{2}\sum_i h_i \log\left(\frac{h_i}{g_i}\right) + \frac{1}{2}\sum_i g_i \log\left(\frac{g_i}{h_i}\right), \quad (1)$$

where g_i and h_i are the frequency (probability) of the ith bin in the histogram of I and J ($\sum_i h_i = \sum_i g_i = 1$). If using base 2 logarithms, the divergence lies in the interval [0, 1].

Obviously, MPHs are intractable for large template sizes and for continuous images. To tackle the problem, the authors suggested to cluster the patterns of the image into a number of bins and then record the frequency of clusters in CHPs.

Within-realization variability is defined as average distance between an ensemble of realizations and the TI. Likewise, between-realization variability is defined as average distance between different realizations. By dividing these variability factors to corresponding factors of a reference method and averaging across different scales, relative within- and between-realization variabilities are computed and the final score is defined as the ratio between these two factors.

Although this method is well-accepted within geo-statisticians, it suffers from a number of problems. Inevitably, the method uses a limited size template, and hence, it cannot detect verbatim copies in larger regions, which happens regularly in practice. Furthermore, because it uses square-shaped templates, it is not capable of detecting curved borders between patterns, including those produced using image quilting or graph cut approaches [12, 15]. On the other hand, the relative nature of the method necessitates having access to many reference realizations and makes the output scores dependent on implementation details and parameter setting of the reference algorithm.

Li et al. [15] have proposed an MPS method based on graph cuts, along with a quantitative method to evaluate their own algorithm. While pasting pixels in the simulation grid, their original index in the training image is also recorded in a map called index coherence map [3]. The map is then used to measure the amount of verbatim copy, which is known to be inversely proportional with variability between realizations. The merging index is introduced as the ratio between the number of seamless pieces in the realization to the number of pieces used during simulation. Low values for merging index show that a large number of pieces are merged together to form larger pieces, indicating significant verbatim copy. While the approach allows identifying verbatim copy, its use is restricted to only a specific type of patch-based algorithm. Moreover, implementing it requires modifying the simulation code, making it difficult to use for benchmarking. Some methods were also developed for evaluating the consistency of observed samples with the TI [24, 25].

1.2 Overview of the proposed method

In this paper, we introduce an evaluation criterion that combines pattern innovation and consistency criteria. Pattern innovation is assessed by estimating the coherence map, relying only on the output realizations and without assuming any access to the simulation process. This is achieved by keypoint detection and matching using scale-invariant feature transform (SIFT) features [26], which allows detection of arbitrarily large segments with arbitrary-shaped borders. It can be

assessed using only one realization. Our proposed pattern innovation criterion considers not only the number of segments in the coherence map but also their size.

For consistency evaluation, local binary patterns (LBP) are employed, which are simple and effective features tailored for texture analysis [27]. Binary patterns are obtained by comparing each pixel with its circularly symmetric neighbors. Uniform rotation invariant binary patterns encode the curvature of edges in the image. The contrast of edges is also recorded as standard deviation of neighboring pixels. As will be examined in the body of the paper, LBP/std features are more distinctive compared to steerable filters, adaptive filters, and clustering-based histograms and can be effectively used for anomaly detection in the realizations [21, 28, 29]. Both pattern innovation and consistency measures are expressed in absolute terms in the interval [0,1], allowing the methods to be evaluated on their own.

For developing the methods, we first assume that both the training image and the realization are two-dimensional regular grids. Generalization to 3D grids will be discussed in Sect. 4.8.

2 Quantifying pattern innovation

As suggested by Tan et al. [21], an evaluation method needs to measure the consistency of the realizations with the TI and the variability between realizations. A good MPS approach is one that maximizes both consistency and creativity factors. Instead of comparing a number of realizations with each other, in this paper pattern, innovation is evaluated indirectly by estimating and then analyzing the coherence map. Consequently, the innovation factor can be computed using even a single realization.

Consider the image shown in the Fig. 1, which is composed of two large rectangles, each containing four smaller rectangles. If large rectangles are regarded as building elements of the picture, only 2 = 2 images can be constructed using this TI by permuting the elements (one of them is the TI itself). However, by taking small rectangles as elementary blocks, 8 = 40,320 variants of this image can be produced. Then, for quantifying the innovation capability of a simulation method, we can rely on the size of building blocks of each realization.

The main drawback is that most MPS implementations do not record the origin of the pixels during simulation. Furthermore, many MPS methods do not allow recording the origin (the location) of pixels and patterns in the TI, either because pixels are synthesized as superposition of multiple pixels coming from different points of the TI [30–32] or because values are simulated based on a modeled CCDF (conditional cumulative distribution function) [7]. To provide a general evaluation framework, we ignore all the knowledge available from the simulation process and the nature of the generating MPS method and rely only on the output realizations. We estimate the index coherence map by segmenting the realizations using keypoint detection and matching techniques [26], which are described below.

2.1 Keypoint detection and description

For computing the coherence map for a given realization, we need to determine the origin of each pixel in the training image. Point correspondence problems arise in many typical machine vision problems including stereo matching, image stitching, video object tracking, etc [33]. In all aforementioned problems, the same scene is viewed from two different view-

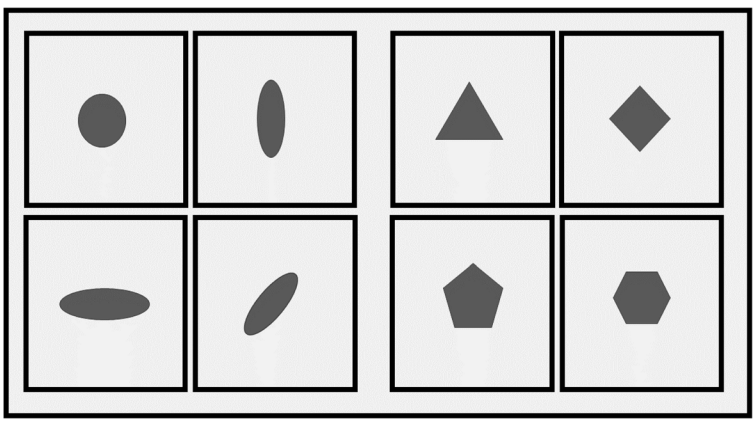


Fig. 1 An example (training) image which is composed of two large rectangles, each containing four smaller rectangles. If large rectangles are regarded as building elements of the image, only two realizations can be derived from this TI. By taking the smaller rectangles as building blocks, we can produce more than 40,000 different realizations

points. Hence, the points will translate in a correlated manner. However, in our problem, the translation vectors for different pixels can be completely independent because nearby parts of the realization can come from pieces of the training image that are far apart.

Consider a realization (*re*) synthesized based on a training image (*TI*). It is impossible to assuredly determine the origin of, say, a black pixel in a flat context among all black pixels in the TI (by origin the location in the *TI* from which the pixel is transferred to the *re*). This holds true even if the point lies on an edge because of the aperture problem [33]. The aperture problem refers to the fact that correspondence problem for a one-dimensional spatial structure, such as an edge, cannot be solved unambiguously if it is viewed through a small aperture.

Instead of considering textureless (flat) regions or pixels on the image edges, people usually rely on keypoints, like corners, to solve the correspondence problem in a pair of images. Different algorithms for keypoint detection in digital images exist in the literature, see for example Lowe [26] or Harris and Stephens [34]. In this paper, we have adopted SIFT (scale invariant feature transform) for keypoint detection and description. The approach is briefly reviewed in the following, and further details can be found in Lowe [26].

At first, the image is convolved with Gaussian filters having different scale factors (σ), and then, the difference between successive filtered images is computed. Candidate keypoints are the local maxima/minima of difference images across scale. To find such points, each pixel is compared to its eight neighbors in the same scale and nine neighbors in each of the neighboring scales. As a result, each detected keypoint is associated with a scale at which the point is detected. Lastly, final keypoints are identified by removing low-contrast points and points along edges. To achieve orientation invariance, a gradient vector is computed for all pixels in the neighborhood around detected keypoints. The gradient of a 2D field $I(x, y)$ is defined as $\nabla I(x, y) = \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right]$. Partial derivatives are approximately computed in discrete images (the simplest approximation for $\frac{\partial I}{\partial x}$ is $I(x+1, y) - I(x, y)$). By forming a 36-bin histogram for gradient vector orientations, the peak orientation of the histogram is identified and assigned to the keypoint (each bin has 10° width and 36 bins cover 360°).

Our goal is to find matches for keypoints of the *re* in the *TI*. This is achieved by extracting robust features from keypoints of both images and then comparing the feature vectors (aka descriptors). In machine vision applications, the description needs to be invariant to photometric changes and changes in orientation and scaling. Although in MPS algorithms, the patterns are often directly pasted to the simulation grid without any changes in orientation, scale, or intensity, some applications apply such transformations to the image, see for example Mariethoz and Kelly [35], Yang et al. [32], Kalantari and Abdollahifard [30], and Pourfard et al. [31].

To form the descriptor vector for each keypoint, at first, the gradient magnitude and orientation are computed for pixels in a window around the keypoint, as shown in the left panel of Fig. 2 [26]. The gradient magnitudes are weighted by a Gaussian window, indicated by a circle in the figure. For each 4×4 sub-region, an 8-bin orientation histogram is formed as shown in the right panel. The value of each bin is determined by accumulating the weighted gradient magnitudes near its corresponding direction. Although Fig. 2 shows a 8×8 neighborhood divided into four sub-regions, in practice, a 16×16 window divided into 16 sub-regions is considered around each keypoint and by extracting 8-bin histograms for each sub-region, a 128 dimensional feature vector is formed. Relying only on histograms of gradient orientations makes the descriptors invariant to photometric changes.

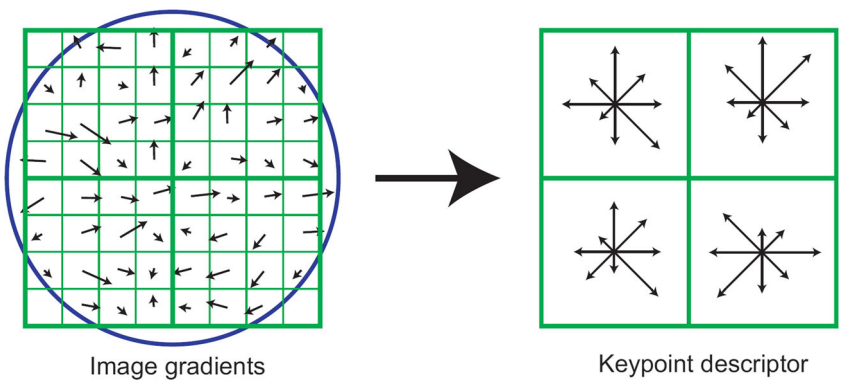


Fig. 2 Keypoint description is done by considering a window around it, partitioning the window into 4×4 sub-regions, extracting 8-bin orientation histogram from each sub-region, and normalizing and concatenating the histograms. The image is taken from Lowe [26]

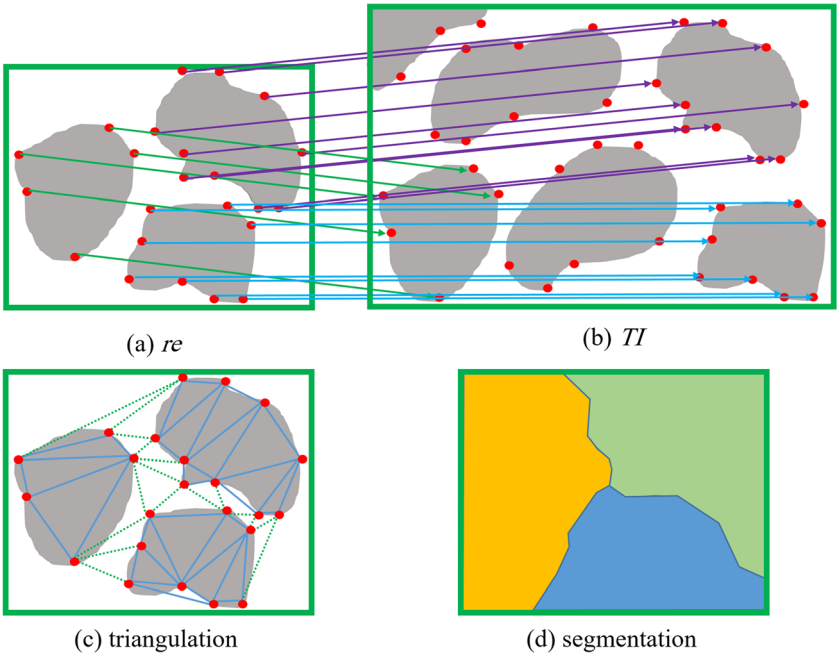


Fig. 3 Image segmentation process: **a** a hypothetical realization obtained based on the TI of **b**. The keypoints in both images are depicted in red dots, and the translation vectors, $\boldsymbol{v}_i$ (i = 1,…,N), are depicted by colored arrows. Similar vectors are shown with the same color. **c** The triangulation, dotted links are to be removed from G to form G'. **d** The final segmentation result

2.2 Image segmentation

Our goal is to partition the realization into separate segments coming from different parts of the TI. Since it is difficult to find matches for flat regions or points on the edge, we only rely on the keypoints. Consider the TI depicted in Fig. 3b and the hypothetical realization of Fig. 3a. The keypoints are independently detected and described for both *re* and *TI*. Let us denote the keypoints of *re* and *TI* by $\boldsymbol{p_i}$ ($i = 1,\ldots, N_p$) and $\boldsymbol{q_j}$ ($j = 1,\ldots, N_q$) and their corresponding descriptor vectors with $\varphi(\boldsymbol{p_i})$ and $\varphi(\boldsymbol{q}_j)$, respectively. Then, each keypoint in *re* is compared to all keypoints in *TI* to find its best match:

$$\boldsymbol{q}_i^* = arg\min_{\boldsymbol{q}_j} \left\| \varphi(\boldsymbol{p}_i) - \varphi\left(\boldsymbol{q}_j\right) \right\|, \tag{2}$$

where ‖.‖ denotes a l_2 norm. It should be noted that at the border of different regions in the *re*, some new keypoints are formed which have no counterpart in the *TI*. To prevent wrong matches, we set an upper limit for the minimum distance between descriptor vectors. If, for a specific keypoint $\boldsymbol{p_i}$, the minimum distance $\min_{\boldsymbol{q}_j} \left\| \varphi(\boldsymbol{p}_i) - \varphi\left(\boldsymbol{q}_j\right) \right\|$ falls above the threshold, t_{des}, the point will be removed from the keypoint set. Considering that all descriptors are normalized so that $\|\varphi(\boldsymbol{p}_i)\| = \|\varphi(\boldsymbol{q}_j)\| = 1$ for all i and j, the threshold is set to 0.2.

If the realization is synthesized with a high degree of pattern innovation or inconsistently with the TI, very limited number of keypoints would pass the aforementioned constraint, leading to unreliable and coarse segmentation results. To prevent excessive point removal, we do not allow removing more than α fraction of keypoints, where α is the constant between zero and one, which we fixed to 0.8. If the fraction of points to be removed is more than α based on the former constraint, we sort the minimum distances and remove α fraction with the highest distances. One may be worried about bad matches that are allowed in this scenario. In practice, such matches are randomly assigned independent and uncorrelated to neighboring keypoints, and as will be discussed later, the algorithm would identify high pattern innovation in such a scenario.

From now on, we denote by $\boldsymbol{p}_i$ the remaining keypoints after removing bad matches, $i = 1,\ldots, N$, and by $\boldsymbol{q}_i^*$ their corresponding keypoints in the *TI*. The translation vector for the ith point, $\boldsymbol{v}_i$, is defined as the vector connecting $\boldsymbol{p}_i$ to its match $\boldsymbol{q}_i^*$:

$$\boldsymbol{v}_i = \boldsymbol{q}_i^* - \boldsymbol{p}_i. \tag{3}$$

The translation vectors are depicted with colored arrows in Fig. 3a and b, and similar vectors are shown with the same color. Keypoints of the realization are triangulated using the well-known Delaunay triangulation method [36] and the resulting graph is denoted by G (see Fig. 3c). Two keypoints, $\boldsymbol{p}_i$ and $\boldsymbol{p}_k$, are called neighbors if they are connected by a link,

e_{ik}, in the graph. Graph G' is formed by removing from G the edges whose endpoints have different translation vectors. In other words, if $\boldsymbol{p}_i$ and $\boldsymbol{p}_k$ are neighbors but $\|\boldsymbol{v}_i - \boldsymbol{v}_k\| > t_{dk}$, then the edge between them, e_{ik}, will be removed from G. The threshold t_{dk} is considered as 1/30 of the realization domain size.

The resulting graph, G', is composed of a set of disjoint subgraphs namely $g_1,\ldots, g_L$ (Fig. 3c). A common label, l, is assigned to all nodes (keypoints) belonging to the same subgraph g_l. The final segmentation is obtained by assigning to each grid node in re the label of its nearest keypoint. Figure 3d shows the final segmentation result. It should be noted that for better visualization in the example of Fig. 3, all contents of the figure including the realization, keypoints and their matches, triangulation, and segmentation are hypothetically drawn by hand. In practice and particularly in the case of continuous images, a large number of keypoints are detected, and subsequently, the segments have smoother borders.

2.3 Segment size distribution

As mentioned before, Li et al. [15] have evaluated the creativity of their method based on only the number of segments found in coherence map, overlooking the size of segments. We believe that for a fair judgment, the size of each segment should be taken into account as well. Uneven distribution of segment sizes can be considered as an indication for lower creativity. Consider a realization made of, say, 10 segments of the TI. If it encompasses a large segment covering 90% of the whole image—while the remaining nine segments cover only 10%—the synthesizing algorithm has very low pattern innovation. On the other hand, if the realization is composed of 10 segments with nearly same size, the innovation can be considered higher. This is commonplace for very structured TIs, where many patch-based algorithms tend to copy a large segment, and at the end of that segment (where there is no more data in the TI), the algorithm tends to copy small segments of it.

The relative size of the lth segment, s_l ($l = 1,\ldots,L$), is defined as the number of pixels in that segment divided by the number of all pixels present in the realization, $\boldsymbol{s} = (s_1,\ldots, s_L)$, $\sum_l s_l = 1$. The l_p norm of the vector $\boldsymbol{s}$ is defined as

$$\|\boldsymbol{s}\|_{l_p} = \sqrt[p]{\sum_{l=1}^{L} s_l{}^p}, \tag{4}$$

Figure 4 shows the vector $\boldsymbol{s}$ under the constraint $\sum_l s_l = 1$ in a two-dimensional space. For $p = 2$, the l_p norm shows the length of the vector. As depicted in Fig. 4 with dashed arrows, the shortest vector corresponds to $s_1 = s_2 = \ldots = s_L$, and the longest vector corresponds to the most uneven situations where one of the size values is one and the remaining values are zero. In general, the norm of the vector $\boldsymbol{s}$ for $p > 1$ could be considered inversely proportional to pattern innovation. Our tests show that by setting $p = 1.3$, the obtained measures are consistent with human judgment.

In the most extreme scenario in terms of pattern innovation, each segment is as small as a single pixel [3]. However, as mentioned earlier in the paper, assuming no access to the simulation process, there is no reliable solution for determining the origin of flat regions in the TI. Furthermore, the origin of a flat region has no practical effect on the realization quality. Therefore, the most innovative is considered as the one in which each keypoint forms a separate segment and all segments have the same size, $\boldsymbol{s}^* = (\frac{1}{N},\frac{1}{N},\ldots,\frac{1}{N})$, where N is the number of keypoints in the realization. The norm of such a vector is

$$\|\boldsymbol{s}^*\|_{l_p} = \sqrt[p]{\sum_{l=1}^{N} \frac{1}{N^p}} = \sqrt[p]{N^{1-p}}. \tag{5}$$

In the least innovative scenario, the realization is composed of a single segment coming directly from the TI

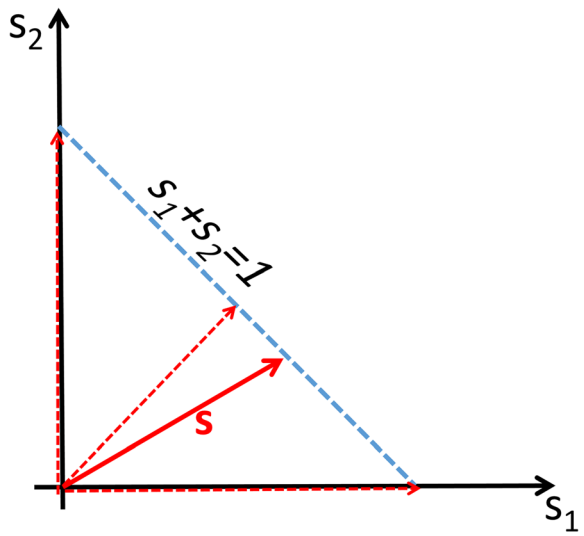


Fig. 4 Depiction of longest and shortest $\boldsymbol{s}$ vectors under the constraint $\sum_l s_l = 1$

($L = 1, s_1 = 1 \Rightarrow \|s^+\|_{l_p} = 1$). We define the innovation factor in the following form so that the innovation factor equals zero and one in the least and most innovative cases respectively:

$$C_i(re, TI) = \frac{\|s\|_{l_p}^{-1} - 1}{N^{\frac{p-1}{p}} - 1} = \frac{\left(\sum_{l=1}^{L} s_l^p\right)^{-\frac{1}{p}} - 1}{N^{\frac{p-1}{p}} - 1}, \quad (6)$$

and $0 \le C_i \le 1$. The algorithm for quantifying pattern innovation is summarized in the following (Algorithm 1).

Algorithm 1 Quantifying the pattern innovation

Inputs : TI and re

Parameters: $p \leftarrow 1.3$, $t_{des} \leftarrow 0.2$, $\alpha \leftarrow 0.8$, $t_{dk} \leftarrow \frac{M}{30}$

- Use SIFT to detect and describe keypoints of TI. Denote the keypoints by $\boldsymbol{q}_j$ $(j = 1, ..., N_q)$ and their descriptors by $\varphi(\boldsymbol{q}_j)$.
- Use SIFT to detect and describe keypoints of re. Denote the keypoints by $\boldsymbol{p}_i$ $(i = 1, ..., N_p)$ and their descriptors by $\varphi(\boldsymbol{p}_i)$.
- Image segmentation:
  - $cnt \leftarrow 0$
  - For $i = 1, \ldots, N_p$
    - Find the best match of $\boldsymbol{p}_i$ in $\{\boldsymbol{q}_j\}$ using equation (2) and denote it by $\boldsymbol{q}_i^*$.
    - $\delta_i \leftarrow \|\varphi(\boldsymbol{p}_i) - \varphi(\boldsymbol{q}_i^*)\|$
    - If $\delta_i \le t_{des}$ then $cnt \leftarrow cnt + 1$
    - $\boldsymbol{v}_i \leftarrow \boldsymbol{q}_i^* - \boldsymbol{p}_i$
  - If $cnt > (1 - \alpha)N_p$
    - Prune $\{\boldsymbol{p}_i\}$ by removing keypoints with $\delta_i > t_{des}$.
  - Else
    - Prune $\{\boldsymbol{p}_i\}$ by removing αN_p points with highest distances (δ_i).
  - Denote by N the number of keypoints remained in the pruned keypoint set $\{\boldsymbol{p}_i\}$.
  - Using Delaunay triangulation, form a graph G with vertices $\{\boldsymbol{p}_i\}$
  - For each link e_{ik} of G that connects p_i to p_k
    - If $\|v_i - v_k\| > t_{dk}$ remove the link
  - Label the keypoints within the connected components of the pruned graph with the same label and assign different labels to disjoint subgraphs.
  - For all nodes of the grid re find the closest keypoint in $\{\boldsymbol{p}_i\}$ and assign the label of that keypoint to the node.
- For $l = 1, \ldots, L$ (for each segment)
  - Compute s_l as the fraction of the nodes with label l to the number of all nodes in re.
- Use equation (6) to compute C_i

3 Quantifying consistency

In this section, we introduce our proposed method for evaluating the consistency of a realization with the TI. For consistency check, we have adopted a histogram comparison framework, just like Heeger and Bergen [28] and Tan et al. [21]. However, in an attempt to find a more distinctive and straightforward approach, we conducted a number of tests as will be discussed briefly in the following, concluding that LBP features can be employed to effectively handle the consistency evaluation problem. Finally, we have introduced a consistency measure based on LBP features.

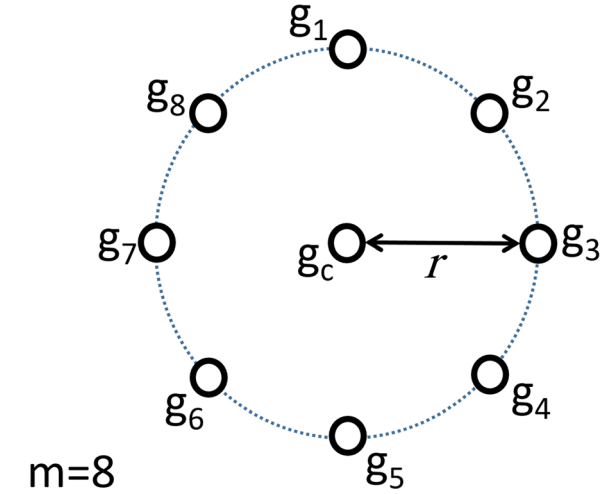


Fig. 5 $LBP_{m,r}$ is computed by comparing the central pixel with m circularly symmetric neighbors at distance r

3.1 Local binary patterns

Local binary patterns (LBP) were initially introduced for texture description and classification [27] and found many applications in machine vision problems due to their simplicity and effectiveness [37–39]. LBP encodes the curvature and contrast of the patterns in an image. It starts by comparing each pixel with its circularly symmetric neighbors and storing the sign of the difference between them.

Consider a central pixel with a gray level g_c and m circularly symmetric neighbors having constant distance r from the center and gray levels $g_1,g_2,\ldots, g_m$ (Fig. 5). The gray level of neighbors lying between pixels is estimated using bilinear interpolation (for categorical variables, nearest neighbor interpolation should be employed). $LBP_{m,\ r}$ is an $m$bit binary number whose ith bit is 1 if $g_i \geq g_c$ and 0 if $g_i < g_c$. Rotation invariant patterns, $LBP^{ri}_{m,r}$, are obtained by circularly shifting the pattern, $LBP_{m,\ r}$, to get the smallest possible number

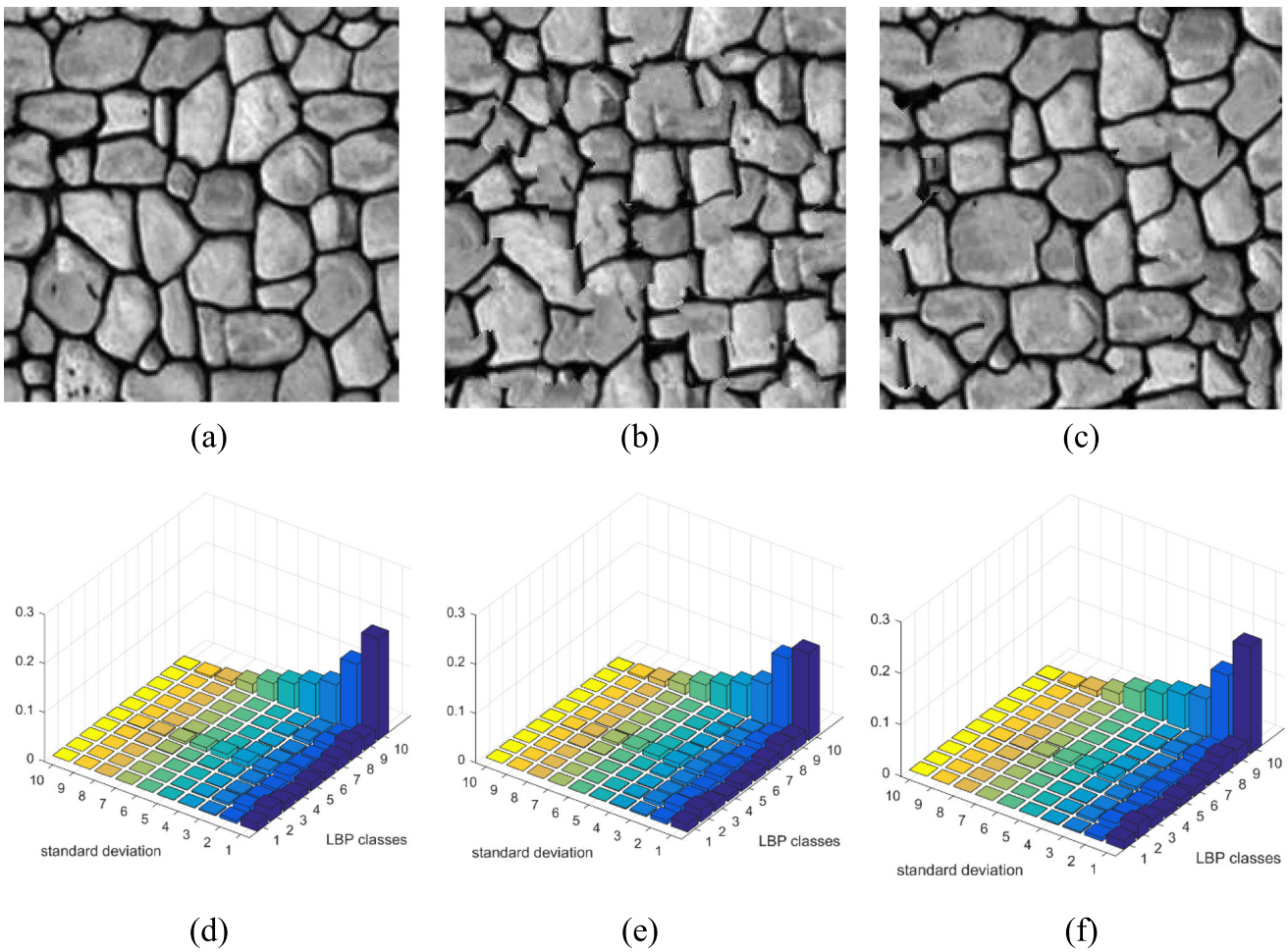


Fig. 6 **a** Stonewall TI, **b**, **c** two realizations obtained by fast image quilting by different parameter settings [40], **d**–**f** corresponding $LBP^{ri}_{8,1}$ /std histograms

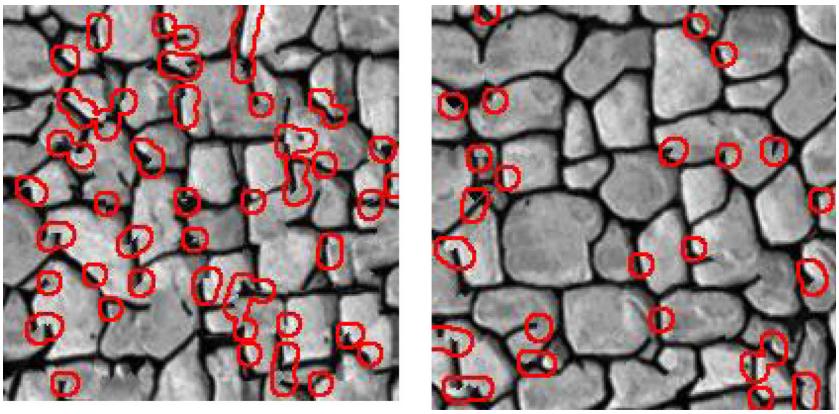


Fig. 7 From left to right, anomaly detection results for Fig. 6b and c

(e.g., for $m = 8$, 11000001 turns to 00000111 by two circular shifts). It should be emphasized that by rotating an image, the histogram of its $LBP^{ri}_{m,r}$ features will remain intact.

Uniform patterns are defined as the patterns with less than three transitions between 0 and 1 while circularly scanning the pattern in a complete round, e.g., 00000000, 00000001, 00000011, …, 01111111, 11111111 for eight bits (for the pattern 00000101, there exist four transitions, and hence, it is non-uniform: 00000´101, 000001`01, 0000010`1, 00000101`). Non-uniform patterns are classified as miscellaneous. For m neighbors, there exist $m + 1$ uniform rotation-invariant patterns (0, $2^1 - 1$, $2^2 - 1$,…, $2^m - 1$) along with a miscellaneous class (total of $m + 2$ classes). 0 represents a bright spot (where the central pixel is brighter than all neighbors), $2^m - 1$ a dark spot, $2^{m/2} - 1$ a straight edge, and other uniform values represent different curves with different degrees of curvature. By forming the histogram of patterns in these $m + 2$ bins, we practically count the number of different curves in the image. Inconsistencies often happen as sharp curvatures along the border between different segments. By using rotation-invariant LBP, we significantly limit the number of LBP classes (bins) and focus on more important features.

Until now, we completely ignored the contrast of each point, which is an important factor in practice. At each point, the standard deviation of the neighboring pixels can also be considered to represent the contrast. By quantizing the standard deviation of each uniform LBP pattern to 10 bins and counting the number of pixels in each bin, 3D LBP/std histograms are formed.

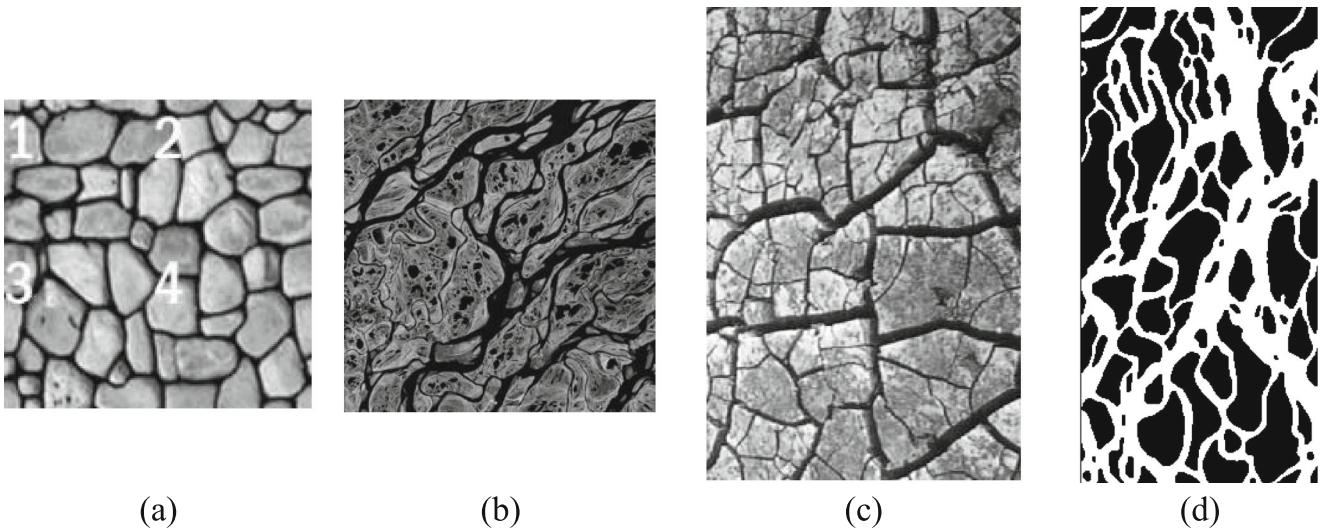


Fig. 8 Some of the TIs employed in our experiments [3], **a** a 200 × 200 image showing a stone wall. **b** A 250 × 250 image showing a piece of a satellite image taken from Lena delta in Russia. **c** A 300 × 200 image showing mud cracks. **d** A 400 × 200 image showing a binary version of a satellite image from the Ganges delta in Bangladesh

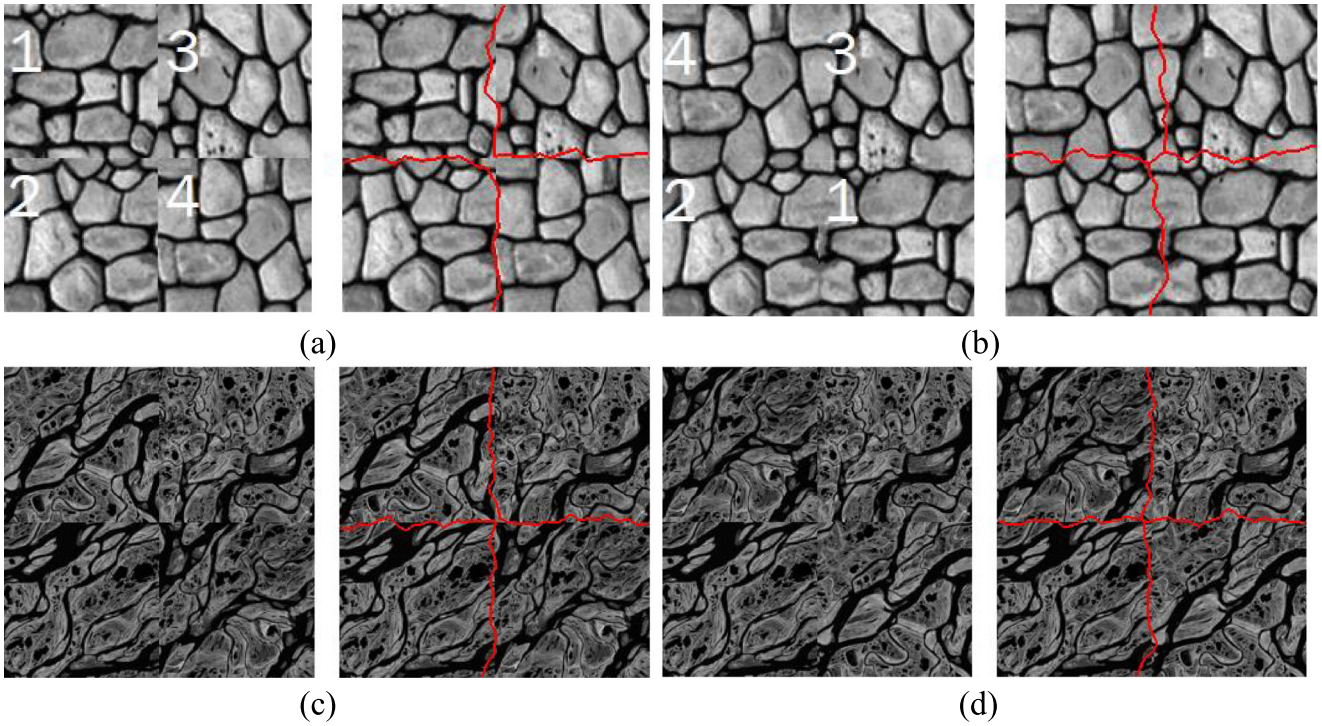


Fig. 9 Four realizations obtained by permuting quarters of the corresponding TIs. **a**, **c** Constructed by switching second quarter for third. **b**, **d** Constructed by switching second for third and first for fourth. The detected segmentation borders are also depicted by red curves

3.2 Histogram-based consistency check

Histogram-based image comparison proceeds by extracting histograms of both images and then comparing them using the Jensen-Shannon divergence of equation (1). Different histograms extracted from images could be employed for this purpose. For example, Heeger and Bergen [28] suggested drawing and then comparing different frequency subbands of the images by applying a number of oriented band-pass filters, a low-pass filter, and a high-pass filter, called steerable filters altogether. Similarly, multiple-point histograms (MPHs) and clustering-based histograms (CHPs) were used for consistency check [21].

Here, we used LBP/std histograms which are known as effective texture descriptors. Figure 6 shows a TI with two realizations obtained using fast image quilting with two different parameter settings [40], along with their LBP/std histograms. Figure 6b and c is obtained by using 2 and 30 DCT (discrete cosine transform) coefficients in the template matching phase, respectively. Visually, the second realization is much more consistent with the TI as compared to the first one. The histograms also confirm this result. Particularly, note the difference in first and second tallest bins. The consistency between images could be quantified by comparing histograms using JS divergence, as will be detailed in next subsection.

Here, we present a short experiment to show the effectiveness of LBP/std features versus steerable filters, adaptive filters, and CHPs. At first, we examined the performance of steerable filters in four different orientations and two different scales along with a high-pass and a low-pass filter [28]. After applying all filters to both *TI* and *re*, the histograms of all filtered images are formed and the average divergence between corresponding sub-bands is computed, as reported in Table 1.

Instead of using fixed steerable filters, we have also tested the effect of adaptive filters obtained based on the TI [29]. After extracting all patterns of a specific size from the TI, nine filters are computed using PCA (principal component analysis) as eigenvectors of the covariance matrix associated with the nine largest eigenvalues.

CHPs and LBPs/std histograms are also extracted from images and the average divergence between the realizations and the TI is computed using different histograms, as shown in Table 1. LBP result is produced as average of 3D histograms associated with $LBP^{ri}_{8,1}$, $LBP^{ri}_{12,2}$, and $LBP^{ri}_{16,3}$.

Table 1 The average divergence, in percent, between TI of Fig. 6a with the realizations of Fig. 6 b and c

	Figure 6b	Figure 6c	Relative divergence
Steerable filters	1.72%	1.33%	1.29
Adaptive filters	1.98%	1.80%	1.10
CHP	6.32%	5.67%	1.11
LBP/std	4.20%	1.94%	2.16

Table 2 Evaluation results on the realizations of Fig. 9. The images contain large segments from the TI and hence, have low creativity and high consistency

	C_i	C_c	C
Figure 9a	0.0825	0.9473	0.0782
Figure 9b	0.1190	0.9969	0.1186
Figure 9c	0.0694	0.9923	0.0688
Figure 9d	0.1003	0.9894	0.0992

The results indicate that, although all aforementioned methods confirm our visual judgment, the scores for two images are relatively close for steerable filters, adaptive filters, and CHP, and the difference between realizations is better highlighted by LBP/std features (note the relative divergence (third column) as the ratio between the numbers in the first and second columns).

Although anomaly detection lies beyond the scope of this paper, here the effectiveness of the LBP/std features is illustrated using a very simple anomaly detection method. At first, the LBP/std histogram of the TI is computed. Then, for each pixel of the realization, LBP/std features are extracted. If the probability of the corresponding bin in the histogram of the TI is below a given threshold, the point is considered as an anomaly. This simple method is applied to Fig. 6b and c and the results are shown in Fig. 7a and b, respectively (the results could be improved by using adaptive thresholds and/or employing the information obtained from the segmentation method). As observed, the detected abnormal features are highly consistent with human judgment. The area of anomalies shows that Fig. 6c is much more consistent with the TI, compared with Fig. 6b. In this experiment, a constant threshold of 2×10^{-4} is applied to the normalized histogram with 10×10 bins of $LBP^{ri}_{8,1}$.

We have also applied $LBP^{ri}_{12,2}$ and $LBP^{ri}_{16,3}$ to the anomaly detection problem, concluding that the best results are obtained using the smallest radius, i.e., $LBP^{ri}_{8,1}$. This is a worthwhile observation indicating that inconsistencies usually occur at fine scales, and looking for inconsistencies in large windows is thus not a good idea.

3.3 Consistency measure

Our proposed consistency measure is developed based on LBP/std features in three different scales, namely $LBP^{ri}_{8,1}$, $LBP^{ri}_{12,2}$, and $LBP^{ri}_{16,3}$. For each (m,r) pair, the corresponding 3D histogram of the realization is computed and compared against the corresponding histogram of the TI using Jensen-Shannon divergence. The average divergence is then computed and denoted by $d_{LBP}(re,TI)$. It should be noted that LBP features cannot be defined for grid points which lie closer than r pixels to the image borders, and hence, such points are excluded from the feature extraction process.

As a rule of thumb in our application, divergences higher than 20% mean that the images are extremely inconsistent (e.g., the divergence between independent images of Fig. 8c and the leftmost image in Fig. 11b is 14.18%). Then, we define the consistency factor as

$$C_c(re, TI) = max(0, 1-5d_{LBP}(re, TI)), \tag{7}$$

where $0 \le C_c \le 1$. For the realizations of Fig. 6b and c, the consistency factor is 0.790 and 0.903, respectively. These absolute values seem to be consistent with human perception. The proposed algorithm for quantifying consistency is summarized in the following Algorithm 2.

Algorithm 2 Quantifying consistency

Inputs : TI and re

- For $(m,r) \in \{(8,1),(12,2),(16,3)\}$,
  - Extract $LBP^{ri}_{m,r}$/std histogram from re and denote it by $h_{m,r}$
  - Extract $LBP^{ri}_{m,r}$/std histogram from TI and denote it by $g_{m,r}$
  - Use equation (1) to compute $d(g_{m,r}, h_{m,r})$
- $d_{LBP} \leftarrow \frac{1}{3}\sum_{(m,r)} d(g_{m,r}, h_{m,r})$
- $C_c\,(re,TI) \leftarrow max(0,1-5d_{LBP}\,(re,TI))$

It is preferable to interpret the obtained innovation and consistency factors independently. We can also combine them in different ways to form a single criterion. The simplest way is to multiply them as follows:

$$C(re, TI) = C_i(re, TI).C_c(re, TI). \tag{8}$$

As will be discussed in the next section, the proposed evaluation criterion is robust in a sense that different realizations

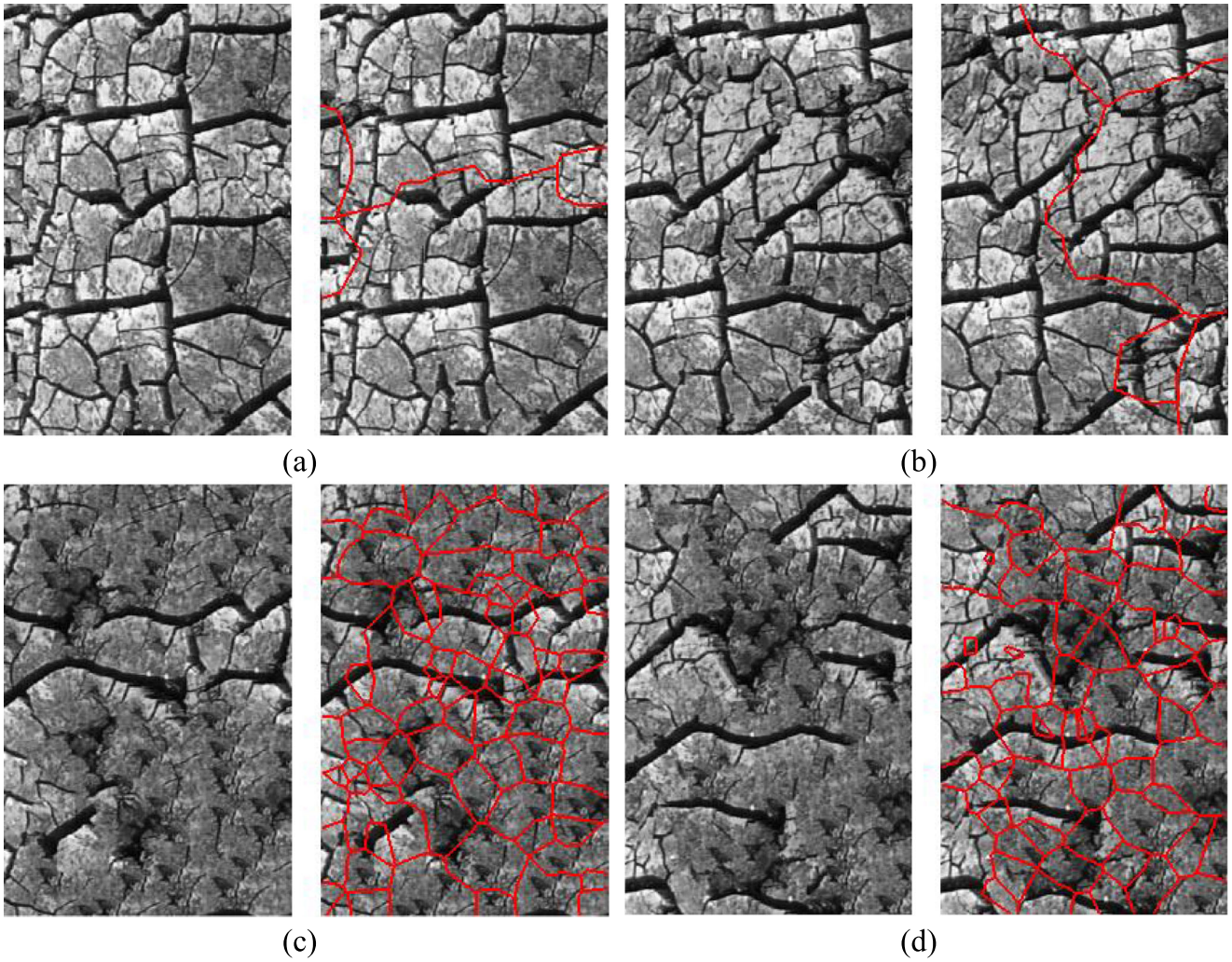


Fig. 10 **a**, **b** Two realizations produced using FPSIM [14], **c**, **d** two realizations obtained from IQ [12]. Template size is 35×35 for both methods, number of replicas N_{rep} is 3 for FPSIM, and 10 for IQ. Images are borrowed from [14], where interested readers can find more details

obtained with the same algorithm and parameters result in similar scores. However, one can improve robustness by averaging over a number of realizations. A better alternative is to compare two different simulation methods, namely A and B, by a relative score (similar to the method of Tan et al. [21]):

$$\gamma^{A,B} = \gamma_i^{A,B} \cdot \gamma_c^{A,B} = \left(\frac{\sum_{k=1}^{K} \left(1 - C_{ik}^{B}\right)}{\sum_{k=1}^{K} \left(1 - C_{ik}^{A}\right)} \right) \cdot \left(\frac{\sum_{k=1}^{K} d_{c,k}^{B}}{\sum_{k=1}^{K} d_{c,k}^{A}} \right), \tag{9}$$

where C_{ik}^{x} is the innovation score for kth realization of the simulation method x and $d_{c,k}^{x}$ is the LBP distance between the same realization and the TI. One advantage of this formulation is that it neutralizes the coefficient used in equation (7). If $\gamma^{A,B} > 1$, it shows that the method A is performing better than the method B and vice versa.

Table 3 Evaluation results of the realizations of Fig. 10

	C_i	C_c	C
Figure 10a	0.0838	0.9314	0.0781
Figure 10b	0.1100	0.9423	0.1036
Figure 10c	0.5986	0	0
Figure 10d	0.5233	0.2423	0.1268

4 Application of the methodology

In this section, the proposed method is applied to realizations simulated using different TIs and simulation algorithms. The

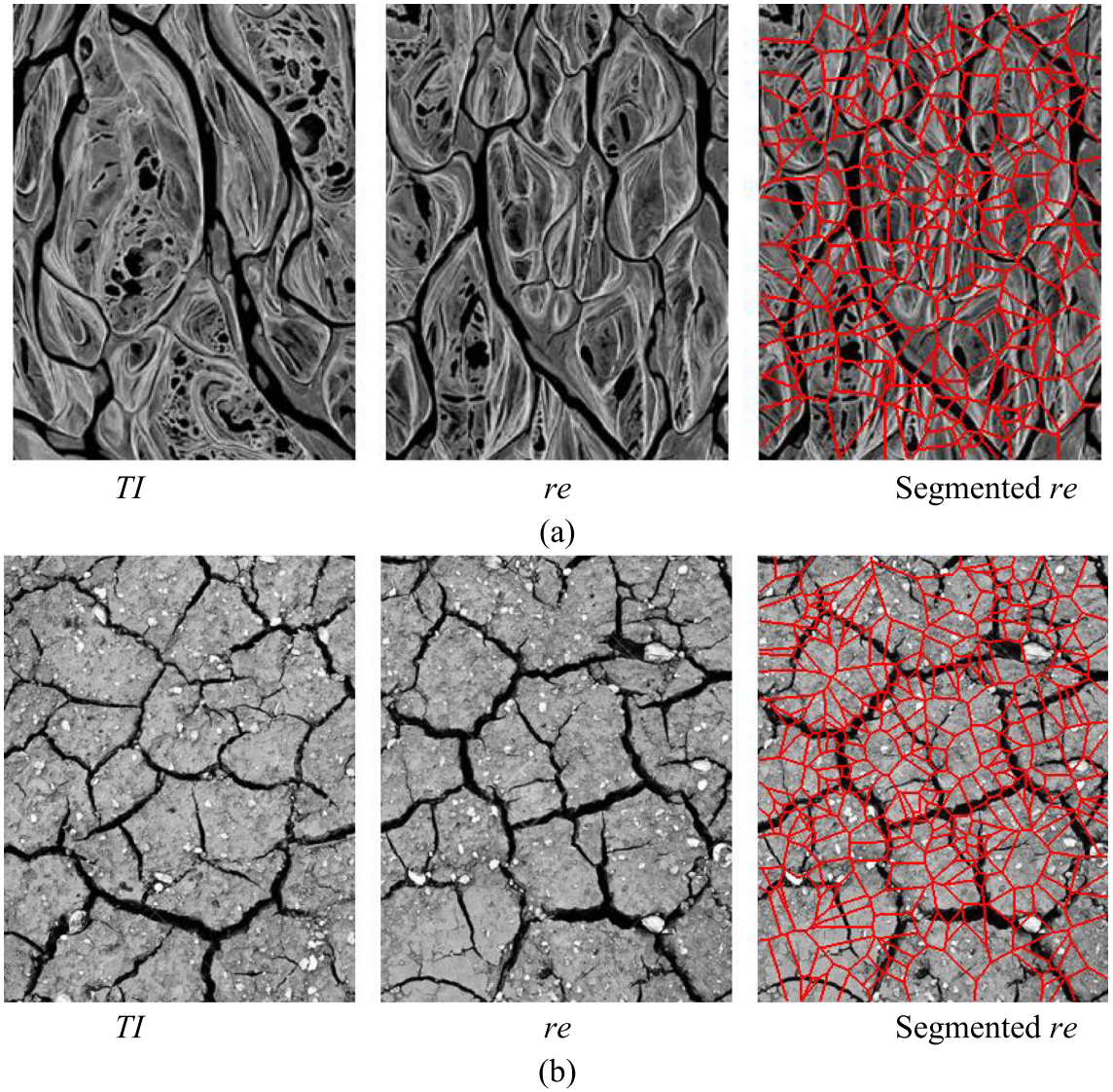


Fig. 11 Two natural images are split in two separate parts, shown in first and second columns, and are considered as TI and *re*, respectively. **a** Satellite images from Lena delta in Russia. **b** Natural textures showing mud cracks

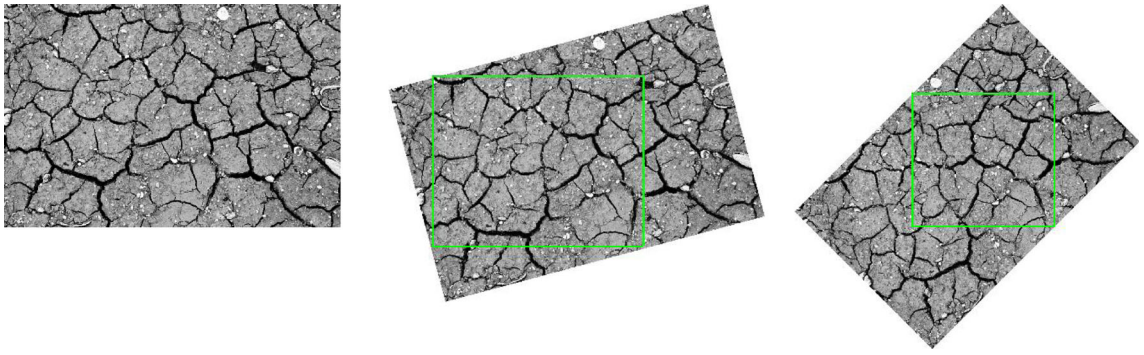
results are presented in the form of quantitative metrics as well as segmentation outputs. Some of the images employed as TIs in our examples are shown in Fig. 8, ranging from simple to very structured textures and binary to continuous.

4.1 Extreme scenarios

We test our method in extreme scenarios. At one end of the spectrum, we consider a low innovation and low consistency case (or in short, LL). Such a scenario is unlikely in practice because if the output realizations of an algorithm are very dissimilar to the TI, the algorithm could be deemed highly innovative (capable of producing new patterns, albeit very dissimilar ones). Other extreme scenarios are represented by LH, HL, and HH, where H stands for high and L stands for low, and the first and second letters describe innovation and consistency, respectively. The HL scenario means that realizations are not consistent with the TI. The LH scenario happens when the realization is composed of large segments of the TI (extreme verbatim copy), with small inconsistencies at the borders of the segments. Usually, there is a trade-off between the two factors, and hence, it is very difficult to produce HH realizations using ordinary simulation algorithms. Here, natural images

Table 4 Evaluation results for Fig. 11

	C_i	C_c	C
Fig. 11a	0.9032	0.9049	0.8173
Fig. 11b	0.8759	0.9504	0.8324



(a) (b) $c_c = 0.9843$ $c_i = 0.0053$ (c) $c_c = 0.9759$ $c_i = 0.0121$

Fig. 12 A reference texture (**a**) along with two versions rotated by 15 and 45°. The computed consistency factors confirm that the rotated versions are completely consistent with the original image

of nearby textures are hypothetically considered as HH realizations.

LH scenario In this section, we apply the evaluation method on realizations with very large segments, constructed by either permuting large square portions of the TI or using previously published simulation algorithms. The TIs of Fig. 8a and b are divided in four quarters. Realizations of Fig. 9a and c are obtained by switching the second quarter for third one and keeping the other two quarters intact. Figure 9b and d is obtained by switching the second quarter for third and first for fourth.

The segmentation borders, estimated using the proposed SIFT-based segmentation method, are depicted by red curves. The location of keypoints depends on the image contents, and hence, they are not distributed regularly along the borders. Consequently, the detected borders are not completely straight. Besides, the segmentation is globally satisfying and in agreement with the known image divisions. In Fig. 9a and c, the first and fourth quarters have similar translation vectors of zero size and there is a small contact between them. Therefore, these images are partitioned into three segments, while the other two images are partitioned into four. However, this has a small impact on innovation factors reported in Table 2. As expected, the algorithm has produced similar small values for innovation factors of all images, due to the detection of large segments.

One may worry about the effect of number of keypoints, N, in equation (6), expecting larger N for more structured images. Although TIs considered in this test are very different in terms of complexity, the innovation factor of their corresponding realizations is not too different. On the other hand, since the realizations are, in fact, copies of the TI, they are highly consistent with it. Only small inconsistencies happen along the borders between quarters, which cover a small portion of the image. For a human observer, the borders between different quarters are much easier to detect in Fig. 9a, compared to all other images. The consistency factors also agree with human judgment, as one can see in Table 2.

Mixed LH and HL The mud crack image (Fig. 8c) is one of the most challenging TIs for all simulation algorithms, containing thick and thin elongated structures (cracks). The thick cracks, covering a small portion of the image, are difficult to match with any pattern other than their adjacent one in the TI. As a result, in any realization, a pattern containing a thick crack is likely followed either with its adjacent pattern in the TI, in which case large segments will be formed (LH), or with a bad match, in which case the realization will be inconsistent with the TI (HL).

FPSIM (fast prioritized simulation) is a patch-based simulation method that highly emphasizes the preservation of pattern continuity by giving priority to high-gradient points [14]. Image quilting is another patch-based method which attempts to improve pattern consistency by cutting patches along minimum error cuts [12]. Two realizations from each of these methods, borrowed from the first paper, are reused in this section (Fig. 10). It should be mentioned that in those examples, the realizations are reloaded after being compressed and stored, which introduces compression artifacts. Next, they are slightly cropped, resized, and then used in our experiments. These slight transformations, which are applied deliberately, make perfect verbatim copy impossible. As will be observed shortly, our method shows acceptable robustness against these modifications thanks to the robustness of SIFT and LBP features.

Once again, although the segmentation results shown in Fig. 10 are not perfect, they seem acceptable in our application. These results imply that the first method suffers from severe verbatim copy problem, and the quantitative factors reported in Table 3 confirm the LH scenario. A human observer can also discover such amount of verbatim copy, but it is not easy to visually detect the borders

between segments. The IQ results suffer from severe inconsistency, which could probably be improved by improved parameterization. The zero consistency reported in the third row means a divergence higher than 20%, which is considered very high in our application. Note that, in the HL scenario, the innovation factor is close to 0.6. In practice, for patch-based methods, achieving higher C_i seems difficult because they are copying patches at least as large as their template size.

HH scenario As mentioned earlier, the HH scenario is not easily achievable using ordinary simulation algorithms. Therefore, in this section, we have used two stationary natural images and split each one in two separate parts, shown in the first and second columns of Fig. 11, and considered them as *TI* and *re*. Since no direct copies exist in the realizations, they are segmented into very small segments, showing high innovation, as confirmed in Table 4. One may expect to see C_i close or even equal to one for these images. However, the non-uniform distribution of keypoints results in segments with slightly different sizes. Therefore, the user should be aware that even for perfect realizations, the algorithm does not produce an innovation factor of one. Furthermore, it should be noted that the number of detected keypoints is significantly higher than the number of segments in the image, but most (possibly $\alpha = 0.8$) of them are removed because the lack of good matches. By considering only the number of remaining keypoints in equation (6), the resulting innovation factor is expectedly high.

4.2 Robustness against rotation

When simulating a nonstationary field, transformed versions of training patterns may be transferred to the SG. In general, LBP features are not invariant against affine or projective transformations, but they are approximately invariant against rotation. Consequently, rotated versions of an image are considered as completely consistent with the original one. Rotation of patches will result in decreased consistency only if it leads to discontinuities or seams. In order to verify this feature, the reference texture of Fig. 12a is rotated by 15 and 45° as depicted in Fig. 12b and c. Then, the regions surrounded by green rectangles are compared against the original texture. The computed consistency factors are 0.9843 and 0.9759 for figures b and c, respectively. Note that apart from the rotation effects, selecting only a sub-region from the image has also contributed in the very small deviations reported (the statistics of a sub-image does not completely match the original image). Therefore, the inconsistency caused by rotation is negligible.

Since nearly all MPS methods complete the simulation grid by only translating patches of the TI, we have not made attempts towards more complex warping models in the segmentation phase (including affine or projective transformations). Nevertheless, the small innovation factors (reported in Fig. 12) show that large verbatim copies are detected in the images. This happens because of tolerating translations smaller than t_{dk}.

4.3 Sensitivity of innovation factor to parameters

While quantifying the innovation factor, four noticeable parameters are used in our algorithm, namely p, α, t_{des}, and t_{dk}. The default values for these parameters are $p = 1.3$, $\alpha = 0.8$, $t_{des} = 0.2$, and $t_{dk} = \frac{1}{30} \times M$, where M is the maximum between the number of rows and columns of the realization. For comparability, it is recommended that users keep these values intact. However, one may be worry about the effect of these parameters.

Our experiments show that apart from p, which directly influences the innovation factor and is selected by trial to produce factors consistent with subjective judgment, the other parameters have limited impacts on the scores. Table 5 shows the sensitivity of the innovation factor for different parameter settings for some realizations of the Sect. 4.1 with respect to the corresponding TIs.

4.4 CPU time

Although no attempts were made towards parallelizing or optimizing the computer code, its required CPU time is acceptable as shown in Table 6. For example, processing two 200 ×

Table 5 The innovation factor for different parameter settings

parameters	α	0.8	0.9	0.7	0.8	0.8	0.8	0.8
	t_{des}	m/30	m/30	m/30	m/20	m/40	m/30	m/30
	t_{dk}	0.2	0.2	0.2	0.2	0.2	0.1	0.3
C_i	Figure 9a	0.0825	0.0825	0.0825	0.0825	0.0825	0.0840	0.0815
	Figure 10b	0.1100	0.1148	0.1023	0.1100	0.1151	0.1100	0.1064
	Figure 10c	0.5986	0.6226	0.6211	0.5673	0.5986	0.5986	0.5986
	Figure 11b	0.8759	0.8565	0.8669	0.8680	0.8816	0.8759	0.8759

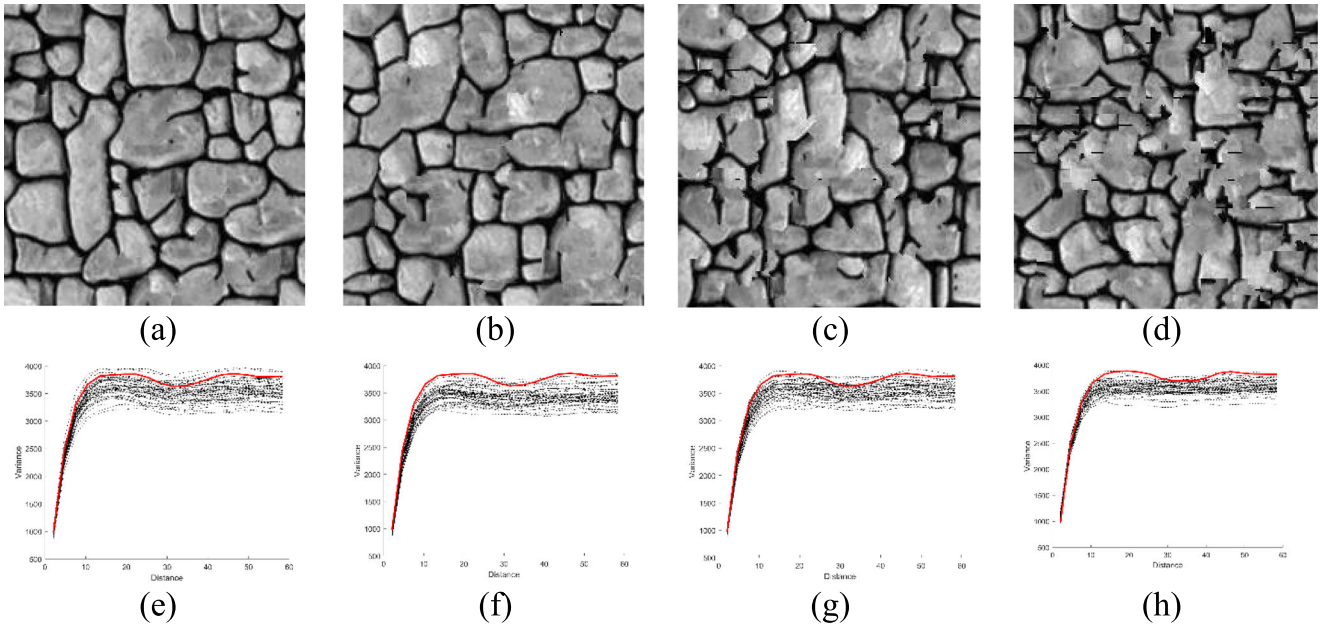


Fig. 13 Conditional simulation results using IQ. Results corresponding to $N_s = 0$, 20, 50,and 100 conditional data are shown in columns 1–4, respectively. First row shows a random realization, and second row shows variograms for 50 realizations along with variogram of the TI (shown with solid line)

200 images takes about 3.25 s. For analyzing a large number of realizations obtained based on the same TI, detection and description of TI keypoints and extraction of LBP features of the TI are done only once. Then, the average time for processing every single 200 × 200 realization tends to 1.6 s. All experiments are performed in MATLAB environment on a laptop computer with Intel(R) Core(TM) i7-6700HQ CPU @2.6 GHz, and 12 GB of RAM.

4.5 Robustness for different realizations

In this section, we have generated 50 unconditional realizations based on each of the TIs in Fig. 8a, b and d, using a version of IQ with a 30 × 30 template size, 7 pixels overlap between subsequent patches, and 20 DCT coefficients for fast query [40]. The mean of the evaluation factors for all 50 realizations is reported in Table 7, along with the standard deviation of C, denoted by σ_C. The standard deviation of the factors shows that the scores have reasonably narrow distributions. There is a high level of stability in the evaluation of realizations produced using a given simulation algorithm and parameterization.

4.6 The effect of conditioning data

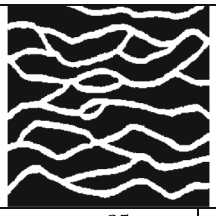
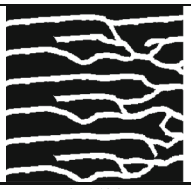
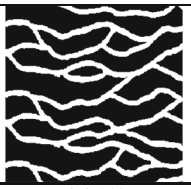
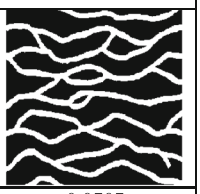
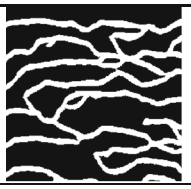
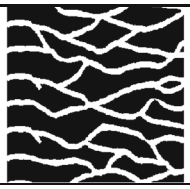
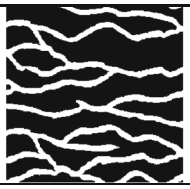
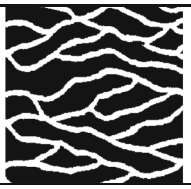
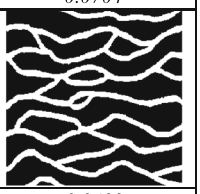
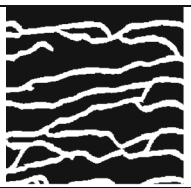
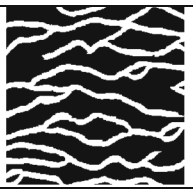
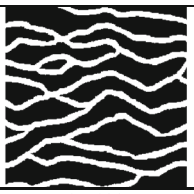
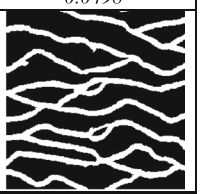
One of the most important and challenging aspects of MPS simulation algorithms is the necessity of satisfying conditional data. Particularly, handling such data is more complicated for patch-based methods as they are pasting a whole patch at once. Since the algorithm can keep the values of conditional data points intact during simulation, its capability for handling conditional data cannot be evaluated by examining only values at these points. Instead, we should study the effect of adding conditional data on the consistency between SG and TI.

In this section, N_s conditional data are considered in the SG (the values are taken from the TI and put in randomized locations in the SG). Then, 50 conditional realizations are generated using IQ [12]. We have repeated this experiment for N_s = 0 (unconditional), 20, 50, and 100. A random realization obtained for each N_s is depicted in Fig. 13 along with

Table 6 CPU time of the proposed method

	TI size	*re* size	# keypoints in *TI*	# keypoints in *re*	Time for C_i (s)	Time for C_c (s)
Figure 9a	200 × 200	200 × 200	798	814	1.36	1.89
Figure 10b	300 × 200	300 × 200	1522	1562	3.38	2.78
Figure 11b	350 × 275	350 × 275	1823	1767	4.39	4.54

Table 9 The realizations produced by IQ and their corresponding scores for different number of replicates N_{rep} and template sizes s_t. Overlap size is taken as $\left\lfloor \frac{s_t}{3} \right\rfloor$. The size of the TI and all of the realizations is 251 × 251

	Training Image: 					
	$s_t = 21$	$s_t = 25$	$s_t = 29$	$s_t = 33$	$s_t = 37$	$s_t = 41$
$N_{rep} = 1$						
C_i	*0.6623*	*0.5450*	*0.5921*	*0.0598*	*0.2661*	*0.0707*
C_c	*0.8718*	*0.9846*	*0.9774*	*0.9951*	*0.9915*	*0.9957*
C	*0.5773*	*0.5366*	*0.5787*	*0.0595*	*0.2639*	*0.0704*
$N_{rep} = 2$						
C_i	*0.8591*	*0.7337*	*0.7149*	*0.6715*	*0.6109*	*0.0499*
C_c	*0.9709*	*0.9745*	*0.9810*	*0.9705*	*0.9918*	*0.9980*
C	*0.8341*	*0.7150*	*0.7013*	*0.6517*	*0.6059*	*0.0498*
$N_{rep} = 5$						
C_i	*0.8625*	*0.8662*	*0.8363*	*0.8377*	*0.7182*	*0.6711*
C_c	*0.9306*	*0.9589*	*0.9816*	*0.9703*	*0.9863*	*0.9838*
C	*0.8026*	*0.8306*	*0.8209*	*0.8129*	*0.7084*	*0.6603*

corresponding variograms of all 50 realizations. Visually, one can confirm that by increasing N_s, the quality of produced realizations is degraded significantly. However, variograms do not exhibit clear changes. Average consistency factors computed using our method are reported in Table 8 along with standard deviations. The values confirm a clear degradation in the quality of the realizations.

4.7 Potential application for parameter setting in MPS algorithms

One of the important challenges of MPS simulation algorithms is how to set their parameters. This is usually accomplished by subjective judgment about the quality of the produced realizations. Although application of the proposed method for solving this problem is not the focus of this paper, here we show a simple example.

Table 7 Average factors for 50 realizations using different TIs

TI	$\overline{C}_i$	$\overline{C}_c$	$\overline{C}$	σ_C
Stone wall	0.5680	0.8882	0.5043	0.0447
Lena delta	0.5314	0.8008	0.4237	0.0447
Ganges delta	0.7113	0.9496	0.6755	0.0312

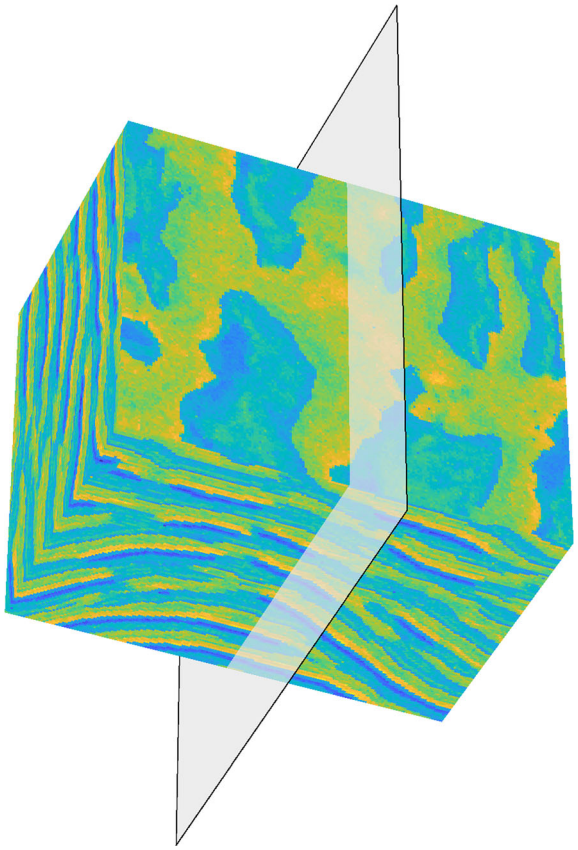


Fig. 14 A 3D grid of size 180 × 150 × 120 showing geological folds [3]. The consistency factor between right and left side of the grid is C_c = 0.9779 and the innovation factor is C_i = 0.7951

Here, IQ method is employed for simulating unconditional realizations based on the channelized TI shown in the first row of Table 9. The effect of two parameters of IQ is examined namely number of replicates N_{rep} and template size s_t. In match-finding phase, IQ finds N_{rep} best matches for a given data-event and selects one of them randomly to paste into the simulation grid. Increasing N_{rep} increases the innovation but has an adverse effect on consistency. By using larger templates (higher s_t), the realizations will be formed from larger blocks of the TI, and therefore, pattern innovation decreases. The overlap size is taken as $\lfloor s_t/3 \rfloor$ in all realizations. It should be noted that, in the IQ implementation used here, the algorithm starts the realizations identically in the top-left corner.

Here, we have generated only one realization for each setting for $s_t \in \{21,25,29,33,37,41\}$ and $N_p \in \{1, 2, 5\}$ as depicted in Table 9. More reliable judgment can be achieved based on a number of realizations for each setting. A human observer usually judges mainly based on discontinuities and ignores the verbatim copies largely. However, the scores produced by the proposed method consider both innovation and consistency factors. The produced scores follow the expected trends. Future work can be devoted to a better combination of these two factors for this application.

4.8 Three-dimensional variables

MPS simulation algorithms have found many two-dimensional applications including, but not limited to, filling missing parts of satellite images, interpolating hyperspectral images, and prediction of rainfall patterns. Many applications are associated with three-dimensional subsurface variables. Nevertheless, because of the difficulties in visualizing 3D variables, visual evaluation of MPS methods is usually done based on 2D realizations.

The proposed quantitative evaluation method can be generalized to 3D variables in different ways. For each pixel in a 2D image, the LBP pattern depends on the order of neighboring pixels. Defining a specific ordering for neighboring voxels in 3D grids in a way consistent with uniformity concept is quite challenging. This can be avoided by computing LBP features from all planes in three perpendicular directions. The histograms of all planes parallel to *xy* plane are accumulated in a single histogram. The same is done for planes parallel to *xz* and *yz* planes forming three separate histograms for $LBP_{8,\,1}$. Repeating this for $LBP_{12,\,2}$ and $LBP_{16,\,3}$, a total of nine histograms are formed for each 3D grid. The average divergence between these histograms and their corresponding histograms in the other 3D grid is considered as d_{LBP} in equation 7.

Keypoint detection and matching are also extensible to three dimensions. Among different implementations of 3D SIFT, we have used the version developed by Blaine Rister (https://github.com/bbrister). After detecting and matching the keypoints in both TI and realization, removing bad matches, triangulating remaining keypoints, pruning the graph, and segmenting the grid into volumetric partitions, we compute the innovation factor using equation (6). Here, the same parameters are used for 3D examples as for previously shown 2D cases.

Consider the 3D grid of Fig. 14 showing geological folds [3]. The grid is partitioned in two equal halves along its longest dimension. Using the mentioned strategy, the consistency

Table 8 The average consistency factor for conditional realizations obtained based on TI of Fig. 6a for different number of conditional data (N_s). The realizations are produced by IQ using 25 × 25 tiles with overlap = 7

N_s	0	20	50	100
$\overline{C}_c/\sigma_{C_c}$	0.93/0.02	0.89/0.02	0.70/0.04	0.28/0.08

between both halves is computed as $C_c = 0.9779$, confirming that they are highly consistent. Innovation factor between the two halves is also computed as $C_i = 0.7951$. These results are in agreement with those reported for the 2D images of Fig. 11.

In another experiment, we set aside the right half and partitioned the left half (the TI) in two separate quarters and formed a new 3D model by switching these two quarters. This new 3D model is then compared with the original left half of the image, leading to $C_c = 0.9954$ and $C_i = 0.0460$. These results are in agreement with those reported for Fig. 9. The obtained results suggest that the method is equivalently applicable to 3D variables.

5 Practical guides

Evaluating realizations obtained based on a training image is a challenging problem. We have tested the proposed method in different scenarios concluding that the results are often reliable. However, we have noticed some practical points during our numerous tests that can be useful for the users:

1.Number of keypoints: the ratio of the number of keypoints to the number of all grid nodes increases proportional with the complexity of the image texture. For continuous images of Figures 9, 10, and 11, this ratio is 1.9–2.5% (see Table 9), and for the simple binary image of Table 9 which has smooth contours, this ratio is about 0.3%. If this ratio decreases excessively (e.g., below 0.3%), the segmentation results tend to be unreliable. This may occur for images with very simple textures.

2.Pixel-based simulation methods: pixel-based simulation methods fill the simulation grid pixel by pixel. At final stages of simulation, such methods encounter excessive constraints (lots of conditioning data), and hence, they rarely can find suitable matches in the TI. This usually leads to short-range inconsistencies and distorted contours. Since LBP is highly sensitive to short range inconsistencies, it penalizes these distortions greatly and tends to produce low consistency (and high innovation) factors. For these methods, we suggest evaluating the realizations relative to the realizations of a reference method of the same category (use equation (9)).

3.Categorical images: for categorical images, one may worry about the validity of comparisons between labels of the pixels in a neighborhood for finding LBP (e.g., if we show *clay* with 1 and *sand* with 2, can we compare clay with sand: *sand* > *clay*?). Although such comparisons are not meaningful, if we preserve consistency and show the same facies with the same labels in both *TI* and *re*, the produced scores are reliable.

4.Combining consistency and innovation factors: combining consistency and innovation factors to a single score (by simply multiplying them) is not a good idea. We suggest interpreting these factors independently.

5.Relative scores: although the absolute scores presented all over this paper seem reliable and trustworthy, we encourage the users to use relative scores of equation (9) to compare a bunch of realizations of a method with a well-accepted reference method of the same category. This strategy will increase the reliability of the scores and is particularly suggested where the absolute scores are too small.

6 Conclusion

In this paper, a quantitative evaluation criterion for multiple-point simulations is introduced by combining pattern innovation and consistency factors. The presented scores are absolute values in the interval [0,1], allowing independent evaluation of MPS realizations. Consistency of the realizations with the TI is evaluated using effective feature descriptors, namely LBP, which show more distinctiveness compared to steerable filters, adaptive filters, and CHPs. By segmenting the realization into seamless patterns copied directly from the TI and then incorporating the segment size distribution into an innovation factor, we have assessed the capability of the simulation method for generating diverse realizations. Experiments in different scenarios show that the reported factors are in agreement with human perception. The user of the method should be aware of the following points. First, the consistency evaluation criteria is rotation invariant, i.e., a rotated version of an image will be identified as completely consistent with the original one. Second, divergences above 20% are considered as extreme inconsistency ($C_c = 0$). Third, even for perfect realizations, the innovation factor is less than (and close to) one. It should also be noted that if the realizations and/or TIs are very simple and smooth, the method do not produce enough keypoints and the innovation factor is rather unreliable. In presented experiments, the final criterion shows small deviations for realizations obtained using a single simulation method with the same parameterization. This would minimize the need to produce many realizations for evaluation of the method.

Acknowledgments The authors would like to express their sincere thanks to anonymous reviewers who devoted their time and expertise to improving this paper. Their comments helped us in improving the paper presentation and proposed formulations.

References

1. Mariethoz, G., Lefebvre, S.: Bridges between multiple-point geostatistics and texture synthesis: Review and guidelines for future research. Comput. Geosci. **66**, 66–80 (2014)
2. Mariethoz, G., Linde, N., Jougnot, D., Rezaee, H.: Feature-preserving interpolation and filtering of environmental time series. Environ. Model Softw. **72**, 71–76 (2015)

3. Mariethoz, G., Caers, J.: Multiple-point geostatistics: stochastic modeling with training images. John Wiley & Sons, Hoboken (2014)
4. Guardiano, F.B., Srivastava, R.M.: Multivariate geostatistics: beyond bivariate moments. In: Geostatistics Troia'92, pp. 133–144. Springer, Berlin (1993)
5. Abdollahifard, M.J., Faez, K.: Fast direct sampling for multiple-point stochastic simulation. Arab. J. Geosci. **7**, 1927–1939 (2014)
6. Mariethoz, G., Renard, P., Straubhaar, J.: The direct sampling method to perform multiple-point geostatistical simulations. Water Resour. Res. **46**, (2010)
7. Strebelle, S.: Conditional simulation of complex geological structures using multiple-point statistics. Math. Geol. **34**, 1–21 (2002)
8. Boucher, A.: Considering complex training images with search tree partitioning. Comput. Geosci. **35**, 1151–1158 (2009)
9. Huysmans, M., Dassargues, A.: Direct multiple-point geostatistical simulation of edge properties for modeling thin irregularly shaped surfaces. Math. Geosci. **43**, 521 (2011)
10. Mariethoz, G., Straubhaar, J., Renard, P., Chugunova, T., Biver, P.: Constraining distance-based multipoint simulations to proportions and trends. Environ. Model Softw. **72**, 184–197 (2015)
11. Honarkhah, M., Caers, J.: Stochastic simulation of patterns using distance-based pattern modeling. Math. Geosci. **42**, 487–517 (2010)
12. Mahmud, K., Mariethoz, G., Caers, J., Tahmasebi, P., Baker, A.: Simulation of Earth textures by conditional image quilting. Water Resour. Res. **50**, 3088–3107 (2014)
13. Tahmasebi, P., Sahimi, M., Caers, J.: MS-CCSIM: accelerating pattern-based geostatistical simulation of categorical variables using a multi-scale search in Fourier space. Comput. Geosci. **67**, 75–88 (2014)
14. Abdollahifard, M.J.: Fast multiple-point simulation using a data-driven path and an efficient gradient-based search. Comput. Geosci. **86**, 64–74 (2016)
15. Li, X., Mariethoz, G., Lu, D., Linde, N.: Patch-based iterative conditional geostatistical simulation using graph cuts. Water Resour. Res. **52**, 6297–6320 (2016)
16. Parra, A., Ortiz, J.M.: Adapting a texture synthesis algorithm for conditional multiple point geostatistical simulation. Stoch. Env. Res. Risk A. **25**, 1101–1111 (2011)
17. Abdollahifard, M.J., Faez, K.: Stochastic simulation of patterns using Bayesian pattern modeling. Comput. Geosci. **17**, 99–116 (2013)
18. Tahmasebi, P., Hezarkhani, A., Sahimi, M.: Multiple-point geostatistical modeling based on the cross-correlation functions. Comput. Geosci. **16**, 779–797 (2012)
19. Rezaee, H., Marcotte, D., Tahmasebi, P., Saucier, A.: Multiple-point geostatistical simulation using enriched pattern databases. Stoch. Env. Res. Risk A. **29**, 893–913 (2015)
20. Abdollahifard, M.J., Ahmadi, S.: Reconstruction of binary geological images using analytical edge and object models. Comput. Geosci. **89**, 239–251 (2016)
21. Tan, X., Tahmasebi, P., Caers, J.: Comparing training-image based algorithms using an analysis of distance. Math. Geosci. **46**, 149–169 (2014)
22. Lange, K., Frydendall, J., Cordua, K.S., Hansen, T.M., Melnikova, Y., Mosegaard, K.: A frequency matching method: solving inverse problems by use of geologically realistic prior information. Math. Geosci. **44**, 783–803 (2012)
23. Endres, D.M., Schindelin, J.E.: A new metric for probability distributions. In: IEEE Transactions on Information theory (2003)
24. Abdollahifard, M.J., Baharvand, M., Mariéthoz, G.: Efficient training image selection for multiple-point geostatistics via analysis of contours. Comput. Geosci. **128**, 41–50 (2019)
25. Pérez, C., Mariethoz, G., Ortiz, J.M.: Verifying the high-order consistency of training images with data for multiple-point geostatistics. Comput. Geosci. **70**, 190–205 (2014)
26. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**, 91–110 (2004)
27. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. **24**, 971–987 (2002)
28. Heeger, D.J., Bergen, J.R.: Pyramid-based texture analysis/synthesis. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 229–238. ACM, New York (1995)
29. Sharifzadehlari, M., Fathianpour, N., Renard, P., Amirfattahi, R.: Random partitioning and adaptive filters for multiple-point stochastic simulation. Stoch. Env. Res. Risk A. **32**, 1375–1396 (2018)
30. Kalantari, S., Abdollahifard, M.J.: Optimization-based multiple-point geostatistics: A sparse way. Comput. Geosci. **95**, 85–98 (2016)
31. Pourfard, M., Abdollahifard, M.J., Faez, K., Motamedi, S.A., Hosseinian, T.: PCTO-SIM: Multiple-point geostatistical modeling using parallel conditional texture optimization. Comput. Geosci. **102**, 116–138 (2017)
32. Yang, L., Hou, W., Cui, C., Cui, J.: GOSIM: a multi-scale iterative multiple-point statistics algorithm with global optimization. Comput. Geosci. **89**, 57–70 (2016)
33. Szeliski, R.: Computer vision: algorithms and applications. Springer Science & Business Media, Berlin (2010)
34. Harris, C., Stephens, M.: A combined corner and edge detector. In: Alvey vision conference, vol. 50, pp. 10–5244. Citeseer (1988)
35. Mariethoz, G., Kelly, B.F.: Modeling complex geological structures with elementary training images and transform-invariant distances. Water Resour. Res. **47**, (2011)
36. Guibas, L.J., Knuth, D.E., Sharir, M.: Randomized incremental construction of Delaunay and Voronoi diagrams. Algorithmica. **7**, 381–413 (1992)
37. Heikkilä, M., Pietikäinen, M., Schmid, C.: Description of interest regions with local binary patterns. Pattern Recogn. **42**, 425–436 (2009)
38. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: Application to face recognition. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2037–2041 (2006)
39. Guo, Z., Zhang, L., Zhang, D.: A completed modeling of local binary pattern operator for texture classification. IEEE Trans. Image Process. **19**, 1657–1663 (2010)
40. Abdollahifard, M.J., Nasiri, B.: Exploiting transformation-domain sparsity for fast query in multiple-point geostatistics. Comput. Geosci. **21**, 289–299 (2017)