Computers & Geosciences 57 (2013) 44–53

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

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

Multiscale Hessian fracture filtering for the enhancement and segmentation of narrow fractures in 3D image data

Maarten Voorn [a,*], Ulrike Exner [b], Alexander Rath [a,c]

[a] Department of Geodynamics and Sedimentology, University of Vienna, Althanstraße 14, 1090 Vienna, Austria
[b] Natural History Museum, Department of Geology and Palaeontology, Burgring 7, 1010 Vienna, Austria
[c] OMV E&P GmbH, ESG-D Production G&G, Trabrennstraße 6-8, 1020 Vienna, Austria

ARTICLE INFO

Article history:
Received 28 December 2012
Received in revised form
4 March 2013
Accepted 6 March 2013
Available online 26 March 2013

Keywords:
μCT
ImageJ
FIJI
Macro
Dolomite
Fractured reservoir

ABSTRACT

Narrow fractures—or more generally narrow planar features—can be difficult to extract from 3D image datasets, and available methods are often unsuitable or inapplicable. A proper extraction is however in many cases required for visualisation or future processing steps. We use the example of 3D X-ray micro-Computed Tomography (μCT) data of narrow fractures through core samples from a dolomitic hydrocarbon reservoir (Hauptdolomit below the Vienna Basin, Austria). The extraction and eventual binary segmentation of the fractures in these datasets is required for porosity determination and permeability modelling.

In this paper, we present the multiscale Hessian fracture filtering technique for extracting narrow fractures from a 3D image dataset. The second-order information in the Hessian matrix is used to distinguish planar features from the dataset. Different results are obtained for different scales of analysis in the calculation of the Hessian matrix. By combining these various scales of analysis, the final output is multiscale; i.e. narrow fractures of different apertures are detected. The presented technique is implemented and made available as macro code for the multiplatform public domain image processing software ImageJ. Serial processing of blocks of data ensures that full 3D processing of relatively large datasets (example dataset: $1670 \times 1670 \times 1546$ voxels) is possible on a desktop computer. Here, several hours of processing time are required, but interaction is only required in the beginning. Various post-processing steps (calibration, connectivity filtering, and binarisation) can be applied, depending on the goals of research.

The multiscale Hessian fracture filtering technique provides very good results for extracting the narrow fractures in our example dataset, despite several drawbacks inherent to the use of the Hessian matrix. Although we apply the technique on a specific example, the general implementation makes the filter suitable for different types of 3D datasets and different research goals.

1. Introduction

With an increasing availability of imaging techniques in the geosciences, there is an increasing need for methods to extract information from such datasets. Three-dimensional imaging techniques, such as X-ray micro-Computed Tomography (μCT), nuclear magnetic resonance (NMR) tomography or Focussed Ion Beam-Scanning Electron Microscopy (FIB-SEM) tomography, generally provide a lot of data that need to be processed efficiently. As very specific and various problems can be addressed using such data, image processing and image analysis techniques need to be applied accordingly. In many cases, it is desired to define features in a dataset by simplifying the original (greyscale image) information. This is often done by segmentation, where a distinct set of classes is created and every single datapoint is assigned to a class. The simplest example of segmentation is binarisation, where every datapoint is assigned to either background (for example: solid material) or foreground (for example: pores). Such a simplified dataset can then serve as input for future analysis, such as permeability modelling. Besides segmentation, fuzzy classifications or other filtering techniques can be applied to simplify the data or improve visualisation, all depending on the final desired goals with the datasets.

In this paper, we cover the specific problem of narrow fractures—or more generally speaking narrow features with a planar shape—in 3D datasets. In our case, the fractures are of importance for the storage (porosity) and transport (permeability) of gas in a fractured dolomitic hydrocarbon reservoir (Hauptdolomit) below the Vienna Basin, Austria. From drillcores taken from these rocks

* Corresponding author. Tel.: +43 1 4277 53473.
E-mail addresses: maarten.voorn@univie.ac.at (M. Voorn), ulrike.exner@nhm-wien.ac.at (U. Exner), alexander.rath@omv.com (A. Rath).

0098-3004/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cageo.2013.03.006

at depth, cylindrical sample plugs of 2 and 3 cm in diameter were selected, and scanned using μCT. The details of these scans are not of importance for the objectives of this paper, since we only focus on the extraction of features from a 3D dataset. For a (geoscientific) overview on μCT scanning and important initial image processing steps required, the reader is referred to e.g. Stock (1999), Van Geet et al. (2000), Ketcham and Carlson (2001), Mees et al. (2003) and Cnudde et al. (2006). Since μCT data on rock samples is mainly a representation of material density, fractures are dark, and solid material is brighter, in greyscale images of such data.

The fractures in our datasets often have an aperture of <10 voxels, and various apertures are present throughout every single dataset. With features being that narrow, the partial volume effect (PVE) and blurring affect the reconstructed μCT data, resulting in the fractures showing up brighter ("having a higher density") than they theoretically should have, as empty space. This reduced contrast to the surroundings makes the extraction of these fractures difficult or impossible by global, greyscale based segmentation techniques. An example of simple global thresholding (binarisation by setting a single grey value, above which all voxels are assigned to one material, and below which all voxels are assigned to another material) on one of our μCT datasets is shown in Fig. 1. Clearly, using such a global threshold does not provide satisfying results for such datasets. This approach is hampered even more by other artefacts often present in μCT data, such as beam hardening effects, remnants of ring artefacts, and of course noise.

Because global thresholding techniques are usually not successful on extracting narrow fractures from a dataset, and manual digitising is not practical in such large datasets (billions of voxels), local and/or structure-based techniques are more viable. Various approaches exist in literature to process fractures in 3D (μCT) datasets (e.g. Keller, 1998; Van Geet and Swennen, 2001; Landis et al., 2003; Sellers et al., 2003; Karpyn et al., 2007; Christe, 2009; Ketcham et al., 2010), of which some indeed apply such a more sophisticated technique. However, problems with these techniques are that they often still do not work for narrow fractures as in our example datasets. For example, they are not generally applicable (e.g. only work on one single fracture in a whole dataset), have no published codes or require expensive commercial software packages, require a lot of processing power, or only process data in two dimensions. Because of these limitations, in this paper we present a different approach, based on the Hessian matrix, for processing narrow fractures in 3D datasets. This approach allows us to extract the fractures of various apertures efficiently from our μCT datasets, without the requirement of extensive computational power. The automated implementation is available for use in the multiplatform public domain software ImageJ (Rasband, 2012). Note that although we use the example of μCT data here, the technique can potentially be used for any 3D dataset.

2. Methods and results

2.1. Introduction to feature detection using the Hessian matrix

2.1.1. Hessian matrix characteristics

The Hessian matrix is, for 3D image data, a 3×3 symmetric matrix, containing the second order partial derivatives of the input image data $I(x,y,z)$:

$$H=\begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \quad (1)$$

The Hessian matrix therefore "describes the second order structure of intensity variation around each point of a 3D image" (Sato et al., 1997). Since second order information describes curvature, another way to think about it is that the Hessian matrix represents the local curvature of the data in a small neighbourhood surrounding each voxel (e.g. Descoteaux et al., 2005).

Above interpretations can be better understood when taking the method of determining the elements of the Hessian matrix into account. An often applied approach (e.g. Lorenz et al., 1997; Sato et al., 1997; Frangi et al., 1998), comes from the linear space scale theory (overview in Lindeberg, 1998), stating that the second derivative of an image can be obtained by convolving the original image with the derivatives of Gaussians. For a single element of the Hessian matrix, this comes down to for example (for a point $I(x,y,z)$):

$$I_{xx}=\left(B\cdot\frac{\partial^2}{\partial x^2}G(x,y,z,s)\right)*I(x,y,z) \quad (2)$$

where B is a factor that can be used for normalisation (not important here), s is a factor of scale (see Section 2.1.2), and G represents a Gaussian function. In one dimension, the Gaussian

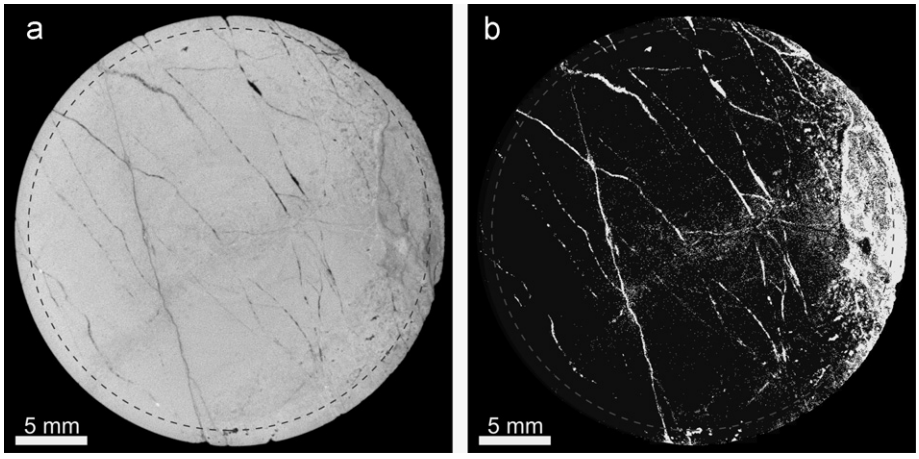

Fig. 1. Input data and binary thresholding. The dashed circle indicates the final region of interest (ROI) as used in the Hessian analysis, to allow comparison to Figs. 3 and 4. Contrast and brightness settings have been altered for display. (a) 2D slice from 3D μCT dataset Prottes TS1, reconstructed at $(18.9\ \mu m)^3$ per voxel. Fractures crosscut a mainly dolomitic host rock. Fracture apertures vary, and larger porous areas can be observed. Material with a lower density, interpreted as more clay rich in hand specimen, is visible on the right hand side of the image. (b) Binarisation of the same slice as in (a) by global thresholding. The here chosen threshold value may be quite high, as especially the right hand side is heavily oversegmented. However, at lower threshold values, undersegmentation becomes a problem in different regions.

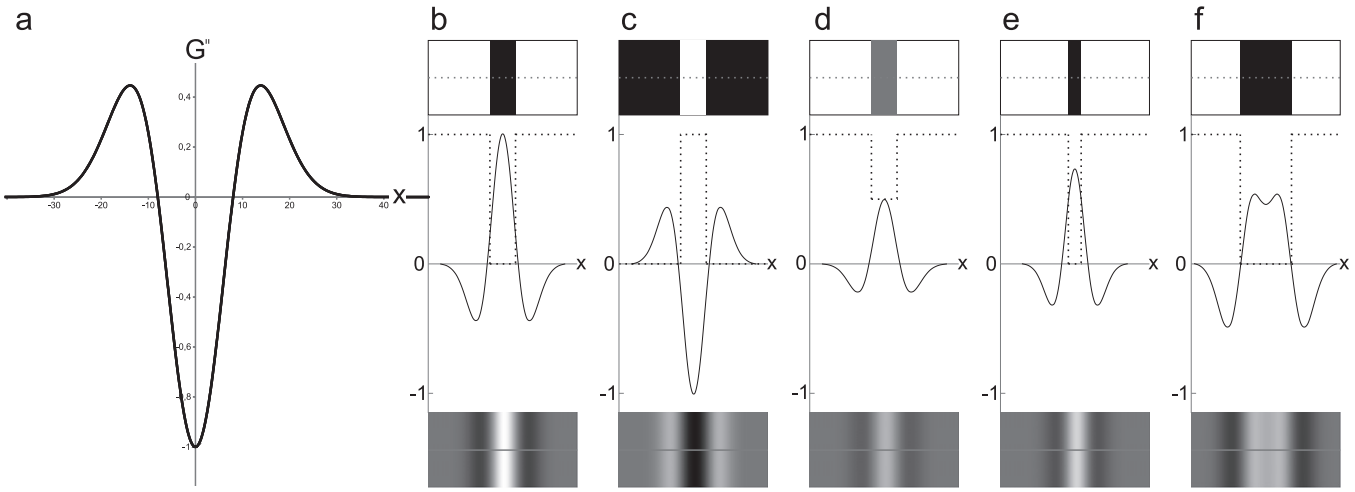

Fig. 2. One-dimensional second derivative of a Gaussian curve, and example results of the convolution to generate one element of the Hessian matrix. In all figures, the same scale setting ($s=8$) is used for the Gaussian. The vertical axes are all normalised between −1 and 1. (a) Second derivative of a Gaussian curve, in one dimension (Eq. (2)). Note the scale of 8 corresponds to the intersection of the curve with the x-axis (at −8 and 8). For the convolution, the best matches for this Gaussian curve will thus be of features of width $2s$ (in this case for a width of 16). (b–f) Top row: input images (duplicated vertically for display). White=1, Black=0. Bottom row: output of the 1-dimensional Gaussian convolution (duplicated vertically for display). White=1, Black=−1. Middle row: graphs displaying the normalised greyscale information of the input data (dashed line) and the output of the 1-dimensional Gaussian convolution (solid line). Note the same normalisations have been carried out for all figures, so the curves and images are directly comparable. (b) Maximum contrast input and maximum response of the convolution, for a feature with a matching width to the Gaussian convolution curve ($2s$; 16 in the example calculation). Dark feature on a bright background. A strong positive response is present for the dark feature. (c) Inverse case of (b). (d) Similar to (b), but for a feature with less contrast to the background (at intensity 0.5). Note the output curve is equally wide as for (b), but has half the intensity. (e) Similar to (b), but for a feature with half the width (s). Note the decrease in intensity of the output, as well as a rather broad representation of the feature: where the two curves in (b) crosscut near the x-axis, the two curves intersect much higher up in this graph. (f) Similar to (b), but for a feature with twice the width ($4s$). Note the decrease in intensity, as well as the dip in the centre of the output curve. This situation shows that for features that are very broad relative to the Gaussian scale, two parallel responses will occur.

function is defined as

$$G(x,s) = C \cdot e^{-\frac{x^2}{2s^2}} \tag{3}$$

C is again a factor that can be used for normalisation (not important here). The second derivative of this Gaussian function is displayed in Fig. 2a, and represents a probe kernel for contrast in the image (Frangi et al., 1998). Simply speaking, when carrying out the convolution of Eq. (2), the image data I is "compared" to the probe kernel (the second derivative of G). Therefore, the highest positive response is recorded in the case of a 100% match. This means, a high positive response is recorded for a dark feature on a brighter background (at the correct orientation and scale) (Fig. 2b). A high negative response is recorded for a bright feature on a darker background (Fig. 2c). Additionally, responses to lesser contrast features (Fig. 2d), narrower features (Fig. 2e) and broader features (Fig. 2f) are displayed for comparison.

When extending this concept to three dimensions, the convolution is carried out in multiple directions, and all elements of the Hessian matrix can be determined. Note that, as emphasised by Lorenz et al. (1997), the second-order nature of the Hessian matrix makes it invariant to grey value offsets, scaling and linear greyscale variations throughout a dataset. This means, only the local contrasts are assessed, whereas the actual greyscale values are not that important. For example, a feature with greyscale 10 on a background of 20, will give the same result as a feature with greyscale value 30 on a background of 40.

After all elements of the Hessian matrix are defined, the eigenvectors and eigenvalues of the matrix can be determined. The eigenvectors of the Hessian matrix describe the local principal directions of curvature, with corresponding eigenvalues λ for their magnitude (Frangi et al., 1998). The eigenvalues thus represent the magnitude of the largest local contrast change, as well as the magnitudes of the local contrast changes in the other two, orthogonal principal directions. The eigenvalue decomposition of the Hessian matrix can therefore be used to distinguish between blob-like, tube-like and plane-like features in a dataset. Table 1 shows the relations between the eigenvalues for such different features. For our narrow fractures with a low intensity on a brighter background, we are interested in the case where one eigenvalue has a high and positive magnitude, and the other two have a small magnitude, as emphasised in Table 1. The eigenvector corresponding to this largest eigenvalue is then the normal to the planar feature.

Table 1
Possible feature patterns for Hessian matrix eigenvalues.

2D		3D			Orientation pattern
λ_1	λ_2	λ_1	λ_2	λ_3	
N	N	N	N	N	Noisy, no orientation pattern
		L	L	H+	Plate-like structure (dark)*
		L	L	H−	Plate-like structure (bright)
L	H+	L	H+	H+	Tubular structure (dark)
L	H−	L	H−	H−	Tubular structure (bright)
H+	H+	H+	H+	H+	Blob-like structure (dark)
H−	H−	H−	H−	H−	Blob-like structure (bright)

$|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$. H=high value; L=low value; N=noise; +/−=positive or negative eigenvalue. Dark=dark feature on a bright background. Bright=bright feature on a dark background. The fracture representation we are interested in is indicated with a *. After Frangi et al. (1998).

2.1.2. Multiscale processing

In the datasets, one should account for features with various apertures. This is possible by altering a parameter when generating the Hessian matrices. As shown in Eqs. (2) and (3) and Fig. 2, the used Gaussian functions introduce a parameter of scale, s. In the Hessian matrix calculation with the plugin FeatureJ, as used in our implementation (see Section 2.2), this is the smoothing scale, defined as the standard deviation of the Gaussian derivative kernels (Meijering, 2010). A certain width scale eventually gives

the highest responses for features with a specific width in the Hessian analysis. The highest responses in any dataset are obtained for maximum contrast features with a width of exactly 2x the applied Gaussian analysis scale (in the FeatureJ definition of scale. Also see Fig. 2). So, with higher smoothing scales, corresponding broader features give a high response in the case of planes. Of course, if one would be looking for blob-like or tube-like features, one would put an emphasis on respectively larger blobs (in 3 directions) and larger diameter (2 directions) tubular features. By combining the results of Hessian analyses on several scales, one gets a good representation of the features over a range of sizes.

2.1.3. Applications of Hessian feature detection

Approaches using the Hessian information of datasets for feature extraction are widespread in the medical sciences (e.g. Lorenz et al., 1997; Sato et al., 1997; Frangi et al., 1998; Descoteaux et al., 2005; Fornaro et al., 2010), but also in other research fields such as material sciences (e.g. Ehrig et al., 2011; Stoessel et al., 2012). The authors have not yet encountered it in geosciences-focussed literature involving feature enhancement or segmentation. The multiscale approach as taken in this paper has its main basis in the medical sciences, where a major application is the detection of (blood) vessels in 3D datasets, as applied in Lorenz et al., (1997), Sato et al., (1997) and Frangi et al., (1998). In their applications, one looks for tubular structures, and the main output for every voxel in the approach of Frangi et al. (1998) is a so-called vesselness value, which essentially indicates the likelihood of the presence of a vessel structure at that point in the dataset. Besides for vessel detection, the multiscale approach has already been adapted for the detection of planar features in 3D datasets, for the detection and segmentation of thin bone structures (Descoteaux et al., 2005). Instead of a vesselness value, a similarly defined sheetness value comes out of this approach. Although this sheetness approach has some similarities with the method presented in our paper, and would most likely provide good results too, our method differs in the order and nature of the calculations performed, and the way of calibration. Our method is characterised by simpler mathematics and later stage calibration. In both the vesselness and the sheetness approaches, the main calibration is introduced by sensitivity parameters defined at the start of the filtering process. Since analysis takes long on large datasets, we choose not to predefine all these sensitivity parameters at the start, and thereby ensure to keep some leeway after the most time consuming processing steps are taken. Additionally, this order of data processing is more suitable for a blockwise serial processing of the datasets.

2.2. Implementation

The hereafter discussed multiscale Hessian fracture filtering method is implemented and made available in macro code, with a graphical user interface (GUI), for the multiplatform public domain image processing software ImageJ (Rasband, 2012), based on Java. For the calculation of the Hessian matrices from the input data, the plugin FeatureJ (Meijering, 2010) is required. This plugin comes packed with the open source software FIJI (Schindelin et al., 2012), a distribution of ImageJ with additional plugins. Note that although ImageJ and FIJI support multi-processor and parallel processing, the Hessian matrix determination in FeatureJ does not.

The Hessian matrix for 3D image data contains 6 independent components for every voxel due to its symmetry. As this means that 7 (input data+6 components), 32 bit floating point numbers for every voxel of input data need to be saved in the RAM memory, memory issues may occur for large datasets on computers with insufficient resources. As we wish for the method to still work on such computers (e.g. desktop PCs), we choose to apply a blockwise serial approach. This means, a suitably-sized (dataset and computer dependent) block of data is read into the RAM memory, on which the 3D calculations are performed, and the results are saved to the hard drive. After this step, the next block is read into the memory, with sufficient overlap to the previous dataset to ensure all final results are truly 3D processed. After all initial Hessian calculations are performed, the data is saved as a series of 2D images. Due to the introduction of so-called control lines (see Section 2.3) to all this processed data, most normalisation, calibration and combination steps after the 3D Hessian calculations can be performed slice-based in 2D. This provides a clear advantage for the speed of processing, while the most important actual fracture detection is performed in 3D. The importance of such 3D processing, and the problems with 2D-only processing of 3D datasets, have been emphasised by for example Elliot and Heck (2007) and Iassonov and Tuller (2010).

The described approach ensures the analysis works on desktop PCs with limited resources (see Section 3.1 for details). It is important to notice that because the intermediate steps are saved to the hard drive, quite some space may be required there. This can go up to several 100s of GBs, depending on the size of the input dataset and the chosen settings. After some initial setup and control steps, the entire filtering process runs without user intervention.

2.3. Multiscale Hessian fracture filtering

The initial step in the multiscale Hessian fracture filtering approach is a preparation step of the data. As input, a stack of 2D images is required, that together make up the 3D dataset. This input data should preferentially be stored as 8 bit or 16 bit grey-scale data. A region of interest (ROI) is defined, and the exterior of this region is set to Not a Number (NaN), so that it does not influence the following filtering steps. At the same time, a set of control lines, with widths in accordance with the desired analysis scales (see Section 2.1.2), is added to every 2D data slice. The contrast of these control lines to their background is calibrated to the contrast of the fractures in the image data by the user. Fig. 3a shows an example of the result of these preparation steps on the data.

After data preparation, the Hessian matrices are determined and the eigenvalues are calculated (output of the FeatureJ filter). The data is processed in the blockwise manner described in Section 2.2. On every block of data, the desired scales of the Hessian matrix are calculated consecutively. The calculations performed on the output eigenvalues are simple. We are only interested in the case of one large positive eigenvalue and two (absolute) small ones (Table 1). By setting the definition of the output eigenvalues to $\lambda_1 \leq \lambda_2 \leq \lambda_3$ (hence NOT absolute as in Table 1), we can use: $A_s = \lambda_{3,s} - |\lambda_{2,s}| - |\lambda_{1,s}|$ if this gives $A_s > 0$; else $A_s = 0$. A is the output intensity, and s represents the different (Gaussian) scales. In this analysis, A_s gets the highest values for the clearest planar features at that scale, that have one single large positive eigenvalue and two near-zero eigenvalues, and takes a consecutively lower value for all less clear planar features.

After the above calculation is performed on every voxel in the dataset for several scales, a normalisation step is carried out. For every image, the following normalisation is performed: $A_s \rightarrow \hat{A}_s$, where $0 \leq \hat{A}_s \leq 1$. This normalisation is done relative to the maximum value in the area with the control lines. This maximum value is always present at the control line of which the width corresponds best to the calculated scale. Throughout the image stack, this value remains constant for every scale. Fig. 3b and c shows the normalised output on two different analysis scales.

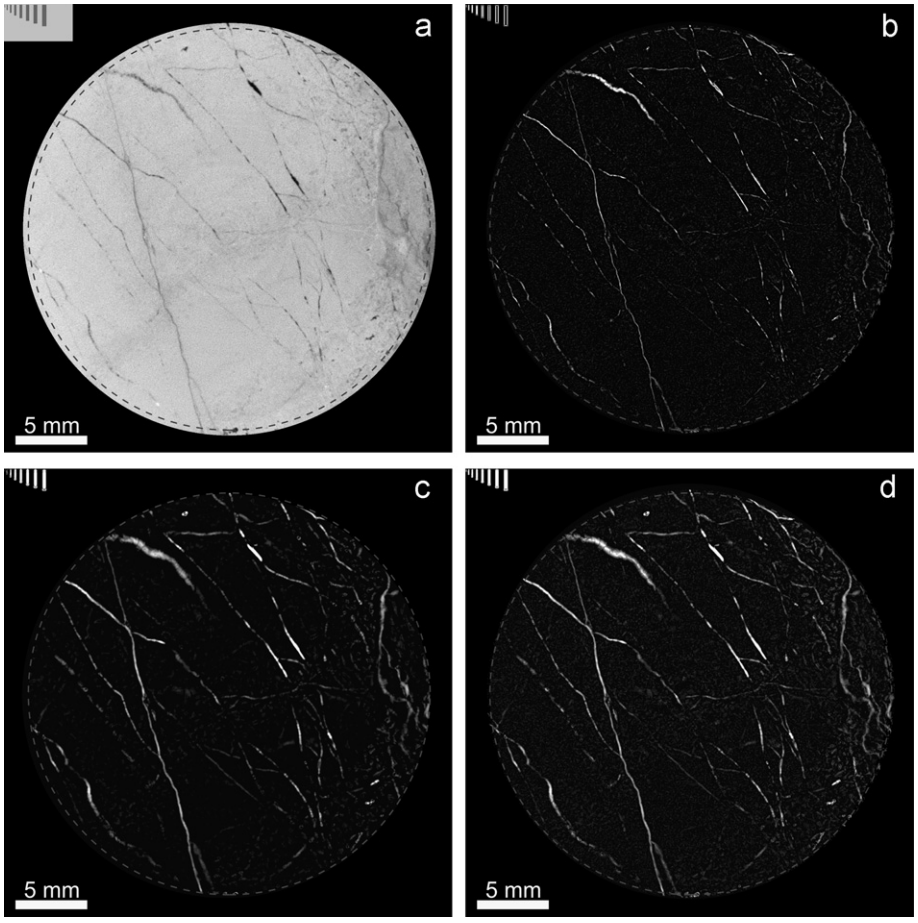

Fig. 3. Steps in the multiscale Hessian fracture filtering procedure. 2D slice representations from the 3D processing procedure. Same slice as displayed in Fig. 1, with the same final ROI indicated by the dashed circle. Contrast and brightness settings have been altered for display. (a) Preparation of the input data before the Hessian filtering. A ROI is defined, where the outside is set to NaN. Note this ROI is larger than the one indicated by the dashed circle, as the final ROI is shrunk to ensure a full 3D output. Control lines are added with a representative contrast for the fractures relative to the surroundings in the image data, for a range of scales. Here, control lines are added for scales 1 to 8, with a stepsize of 1. (b) Normalised results of the combination of the Hessian eigenvalues for fracture enhancement (Section 2.3), for scale 2 (the lowest included in this analysis). Narrow planar features come out clearly, but noise is also apparent. (c) Similar to (b), but for scale 5 (the highest included in this analysis). In comparison to (b), the noise effect is lower, and fractures appear wider. (d) Combined scales 2 to 5, with a stepsize of 1. This image represents the output of the filtering procedure. In our example, the most important planar features stand out clearly, but (low-intensity) noise remains present.

Table 2
Multiscale Hessian fracture filtering input parameters.

Name/ symbol	Parameter	Description
s_{min}	Minimum Gaussian smoothing scale	Relates to minimum aperture to detect. Maximum response for apertures of 2x scale.
s_{max}	Maximum Gaussian smoothing scale	Relates to maximum aperture to detect. Maximum response for apertures of 2x scale.
s_{step}	Steps between Gaussian smoothing scales	Stepsize can be varied relative to the apertures in the data or to speed up the process.
blocksize	Blocksize in amount of slices	Amount of data to be analysed per consecutive step. An optimum can be defined.
avgmat *	Average material greyscale	Greyscale value that characterises the material outside the fractures. Used for control lines.
consthresh *	Conservative threshold of fractures	Greyscale value that estimates the greyscales present in the fractures. Used for control lines.
maxmat *	Maximum material greyscale	Greyscale value that can be used to remove unusually bright spots in the dataset to prevent artefacts in the final output.
padding	Percentage of padding to be performed at start and end of input stack	First and last slices can be copied several times and included in the analysis. Extends the range of output slices but introduces "pseudo-3D" information, increasingly worse at higher percentages. Best kept at zero.

* Parameters that can remain "unspecified" by settings values corresponding to the maximum range of greyscales possible in the dataset (e.g. 0–255 for 8 bit, 0–65355 for 16 bit data). If specified, these values can remain approximate as minor variations do not influence the final result by much. Also note that because of the invariance of the Hessian matrix to the actual greyscale values (only the contrast is of importance), the main specification set by *consthresh* and *avgmat* is the average contrast of fractures to their surroundings. See user guide to the code (Supplementary material) for more details.

After the normalisation, the scales are combined by $\hat{A}=\max(\hat{A}_s)$, for $s_{min}\leq s\leq s_{max}$. This means that for every voxel in the dataset, the maximum response of the Hessian analysis over all scales is taken, thereby combining the various apertures. The result of this is shown in Fig. 3d, and represents the final output of the (initial) filtering.

A complete overview of the parameters required for the multiscale Hessian fracture filtering approach is shown in Table 2.

2.4. Calibration, connectivity, and segmentation

Although the output of the described multiscale Hessian fracture filtering approach may in some cases be directly useable, we apply some additional postprocessing steps for our datasets. The descriptions hereafter are thus rather specific to our data type, and should therefore be seen more as a case study than previous sections. Calibration is performed to better visualise the data and to reduce noise. Additional noise reduction can be achieved by filtering throughout the dataset for connected clusters. Finally, binary segmentation is often required for data processing in a later stage, such as permeability modelling.

2.4.1. Calibration and including a conservative threshold

Calibrating the output of the multiscale Hessian fracture filter can be done in various ways, and is strongly dependent on the desired final output. The calibration we apply results in intensity values from 0 to 100, which can be seen as a fuzzy classification of porosity (e.g. 100=100% porous, 0=0% porous). For noise reduction and for emphasising high intensity values, we take a sigmoid curve based on the cosine function, with a lower and an upper cut-off value (see the online Supplementary material for details). Because such a calibration requires additional user input, it is quite ambiguous, but this is very difficult to prevent in an analysis like the one presented here. Combined with information from different analyses (in our case for example porosity values determined from laboratory measurements or thin section analysis), a better calibration may be achievable. Different calibration curves may also prove more suitable in different analyses.

For the datasets used in our research, an additional step is taken together with the calibration. Besides narrow fractures, also some broader and blob-like porous areas are present in the data. Only the edges of these are detected by the multiscale Hessian fracture filtering. Their internal areas however have very low intensity values, and can easily be selected by a simple global threshold. We therefore select these areas from the original input data, often by using the conservative threshold already defined earlier (Table 2), and include them with the calibration data as 100% porous areas. The combination of this and the calibration is shown in Fig. 4a. Whether such an addition of a conservative threshold is desired by the user is dependent on the input data.

2.4.2. Connectivity filtering

For the examples presented here, the effective porosity and the permeability are eventually of main interest. Because of this, we

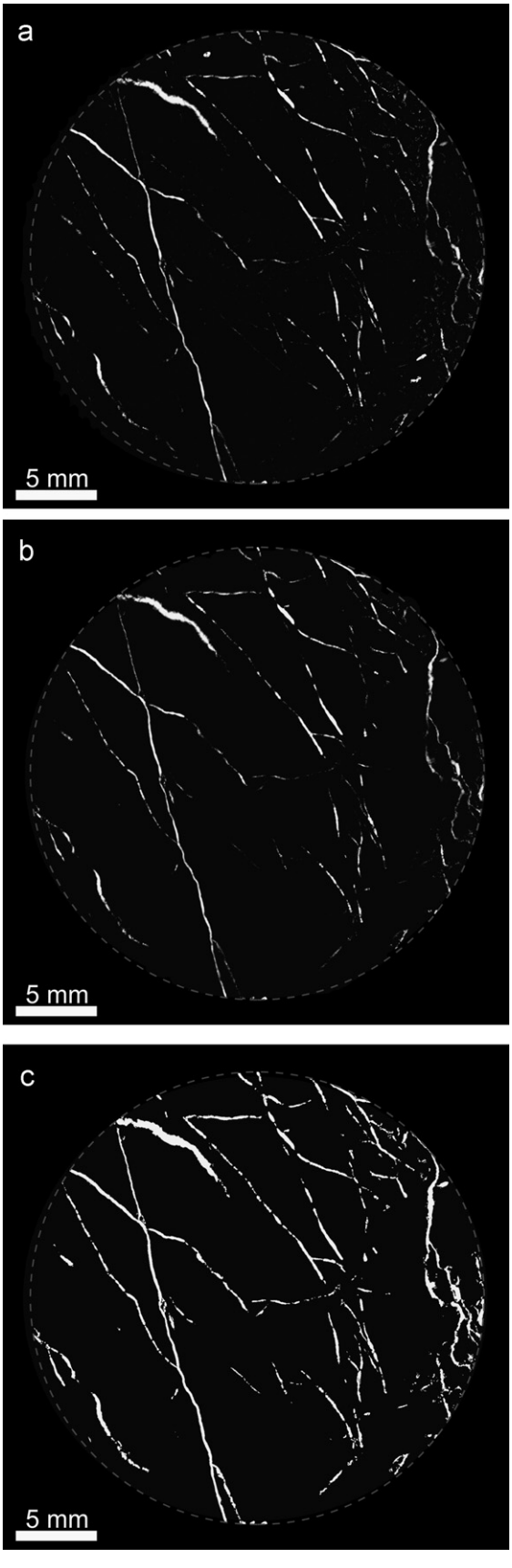

Fig. 4. Calibration and post-processing steps of the data after the multiscale Hessian fracture filtering procedure. The same data is shown as in Figs. 1 and 3, and the same remarks apply. (a) Calibration of the data using a cosine function (sigmoid curve) between two user-selected values (see Supplementary material for details). The calibration shows efficient noise reduction, as well as higher intensities for clear planar features. Additionally, the calibrated result has been combined with a conservative threshold on the original input data, to include larger, non-planar porous areas as well. Output values range from 0 (black) to 100 (white). This can be seen as a fuzzy approach to defining the porosity (for the mentioned output values, a range from 0% to 100% porosity). The performance in the darker area (visible in Fig. 1a) is good (few false positives). Some features appear rather broad in this 2D slice-based representation (e.g. left top corner), due to their inclined nature (better visible in 3D). (b) Same dataset as in (a), but 3D filtered for connected porous clusters (intensities of 1 to 100 here considered as porous) between the first and the last slices of data, using MATLAB® (The Mathworks, Inc., 2011). Compared to (a), some remaining noise and isolated pores are removed, as well as some (parts of) fractures that are apparently not connected to the main porous cluster, as resolvable at this resolution. (c) Binarisation attempt of the data shown in (b). The lowest possible threshold value of 1 is chosen here, explaining why some features may appear too broad. Of main importance is however the comparison to Fig. 1b, clearly demonstrating the better performance of the multiscale Hessian fracture filtering technique over such a simple, direct thresholding approach. For a true aperture analysis, a better calibration and threshold setting may be required, for example by using microscope analyses on thin sections as an input for calibration. In such a case, most likely a higher threshold will be chosen, reducing the apertures of the fractures to more realistic values.

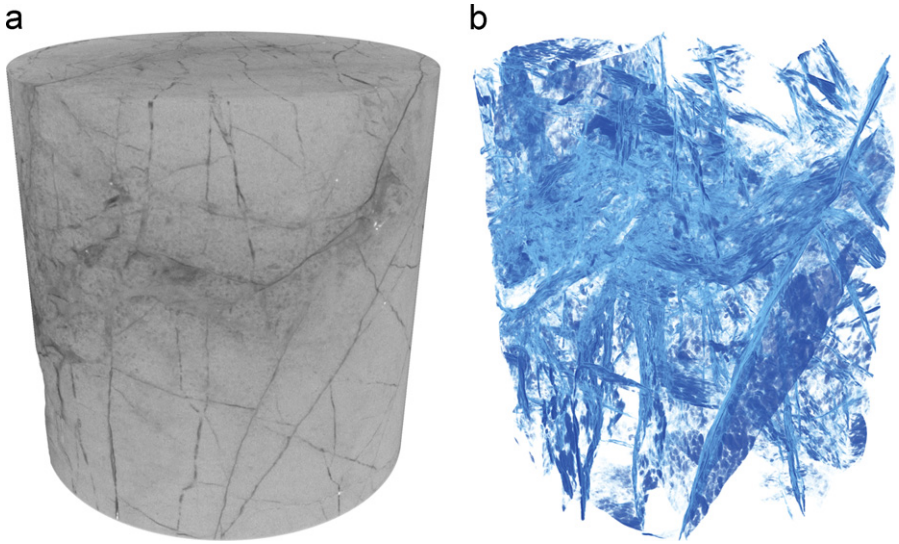

Fig. 5. 2D outtake of a 3D visualisation of input data and the corresponding multiscale Hessian fracture filtering results, created with the visualisation software Amira® (Visualization Sciences Group, 2012). A motion clip, in colour, can be found with the online Supplementary material. Same dataset as in Figs. 1, 3 and 4. (a) Surface rendering of the dataset, with the final ROI applied. This shows the surface representation of the fractures, as well as material variations throughout the dataset. (b) Volume rendering of the fractures in the dataset, in the same orientation as (a). The same steps as for Fig. 4b have been taken (Hessian filtering, calibration and connectivity filtering). The chosen intensities and opacity are mainly for visualisation, but generally speaking, darker areas are more porous. Some fractures are very clear, whereas in other areas the visualisation is less clear. This is usually due to the presence of a lot of closely spaced and crosscutting fractures, other pores, and/or some remaining noise. Gaps in fracture planes mostly represent partial closure of the fracture (or: an aperture below the detection limit for the Hessian filtering), which can be observed in the original raw data as well.

can use connectivity filtering to improve the representation of the data. In this method, we filter for connected porous clusters throughout the sample, in a certain direction. Included with the Supplementary material is a MATLAB® (The Mathworks, Inc., 2011) script, which uses the *bwconncomp* function of the MATLAB® Image Processing Toolbox to determine and label clusters of connected data (26-connected neighbourhoods by default) in the dataset. By comparing the labels of the clusters present in the first and the last image slice, the labels of clusters of data that are connected between these two slices can be selected. By just keeping the clusters with these labels in the entire dataset, only the connected voxels are left, and retain their values from the calibration. All voxels in non-connected clusters are set to be non-porous (0). Generally, noise is effectively removed by this method, but depending on the assessed problem, it may have the undesirable effect of removing some planar features entirely. An example of the calibrated results after connectivity filtering is shown in Fig. 4b, as well as in the 3D visualisation in Fig. 5.

2.4.3. Binary segmentation

Completely binary segmented results may be desired depending on the problem being assessed. There is clearly no single answer on how to do this for every dataset (for comparisons of binarisation techniques, see for example Sezgin and Sankur (2004), Iassonov et al. (2009) and Porter and Wildenschild (2010)), and it is beyond the scope of this paper to go into detail on this. For our data, cross-calibrations with data from different methods may again be of importance here to decide on the best possible obtainable results. An example of binary segmentation on one of our datasets is shown in Fig. 4c.

3. Discussion

3.1. Advantages and drawbacks

The multiscale Hessian fracture filtering technique described in this paper has some very clear advantages for emphasising and eventually extracting narrow fractures in 3D datasets. When comparing Fig. 4c to Fig. 1b, the advantage over global thresholding is evident. The 3D implementation of the Hessian approach ensures that fractures in all orientations can be detected. This is a clear advantage over 2D methods, especially for fractures that are near-parallel to the slicing direction in slice-based datasets.

The full approach is implemented in public domain software and can be run easily, with or without GUI. Some user input is required at the start of the filtering process, which then runs automatically without interruption. Serial processing of the data ensures that even on desktop systems with limited resources, the filtering technique works. The fastest processing time achieved (with optimum settings) for a $1670 \times 1670 \times 1546$ voxels dataset on a desktop with an 8-core Intel Core i7 2600 @ 3.4 GHz processor and 16 GB RAM memory is slightly over 10 h. Note that in fact only one of the processing cores was used. The Hessian matrix was calculated on 6 Gaussian scales, from 2 to 7, and 134 GB of HDD space was required. The same analysis takes over 31 h on a desktop with a 2-core (one core used) Intel Core2Duo E8400 @ 3.0 GHz processor and 4 GB RAM, but this time reduces to a more reasonable 14 h when only taking Gaussian scales 2–5. This displays the large influence of both RAM memory and chosen Gaussian scales. First, the available RAM memory limits the size of the data blocks that can be processed. Second, higher scales take longer to calculate (as then the Gaussian smoothing at every voxel has to account for more surrounding data), and require more overlap between consecutively analysed data blocks. This overlap yields higher memory inefficiency, so more blocks need to be processed eventually. Naturally, the calibration and connectivity filtering steps require some additional processing time, but are separated from the main Hessian analysis.

The described approach requires some user input, depending on the desired outcome and planned future analyses on the dataset. In cases where only a qualitative enhancement of narrow fractures in a dataset is required, the amount of required human input is limited. In this case, next to a user defined ROI, only the Gaussian scales to be analysed need to be chosen. Important to notice is that these scales can influence the filtering results negatively. Including a too low scale can cause noise artefacts to become abundant. A too high scale can lead to an unrealistically

broad representation of the fractures in a dataset. An example can be seen in Fig. 2e. Although the curves displayed in the figure do not represent the full output of the multiscale Hessian fracture filter, it is clear the output of the Gaussian convolution results in a curve broader than the input feature. On the other hand, if not high enough scales are included, very broad features might be outputted as two parallel features (for example, Fig. 2f). In the latter case, this usually will be compensated by including a conservative threshold in the calibration step.

Next to the scales, additional user input may be required. For a more reasonable normalisation of the results to the actual contrasts in a dataset, several greyscale values (*consthresh*, *avgmat* and *maxmat* in Table 2) should be chosen. More user input ambiguity is of course also included by the final calibration and binarisation steps, as well as when turning on data padding (*padding* in Table 2). Although it is always better to reduce human input in image processing to a minimum, there are very few approaches available in literature by which this can be achieved, and especially not for narrow fractures. The described vesselness and sheetness approaches (Frangi et al., 1998; Descoteaux et al., 2005) for example also do not perform better in this sense, as they require sensitivity parameters that control the output to be set by the user.

From Figs. 3 and 4 it is clear that most fractures are detected by our method. An emphasis should however be put on some fractures that are not displayed in the results. The main reason for faint fractures not to be present in the output is noise. For (synthetic) input data without any noise, even the faintest features are detected. As in the calibration no noise reduction is required, these faint features will remain visible. However, for real data, this is often not the case, as it is not possible to retain these features without also retaining a lot of noise. Of course, features can also be so thin, that no Gaussian kernel is appropriate to detect them, without again including a lot of additional noise. Because of the above, it is not possible to pinpoint the exact resolution limits of the presented method, as there is a strong correlation to the quality of the input images.

Another point regarding the fracture detection is that some narrow fractures show up rather broad in the final result. This is especially clear in data thresholded using the lowest possible threshold setting, as in Fig. 4c. This broadening of aperture is of course introduced by having to include Gaussian scales too broad for some features, to successfully detect the truly broader features. This effect is inevitable, but better results can be obtained for better calibrations and threshold settings. For example, a comparison to microscope analyses on thin sections can be very useful in this aspect, to set a better threshold before performing an aperture analysis. Another approach would be to calibrate using data of a fracture with a known and controlled aperture (such an approach has for example been applied in Ketcham et al., 2010). For the example Fig. 4c, this would mean taking a much higher threshold value, to decrease the apertures to more realistic values.

As a final point an important disadvantage, inherent to our Hessian approach, should be mentioned here. Fracture intersections may cause problems in the analysis. The intersection of two fractures is not a planar feature, but tube-like, with the structure being the most tubular when two orthogonal planes intersect. In synthetic data, such an intersection has a value of zero in the results of the Hessian filtering (see Fig. 6). The effect decreases with lower intersection angles of two planes, i.e. the intersection becomes more planar in nature and gives a response higher than 0 in the Hessian fracture filtering results (also see Fig. 6). In our datasets, intersections are often characterised by such lower angles. Additionally, the contrast to the surroundings at intersections is often higher than between single fracture and surroundings, which may increase the response in the Hessian filtering result further. If it contains low enough greyscale values, the intersection may also even be output as porous when the low global threshold on the input data is combined with the calibration results (Section 2.4.1). Therefore, in our data, the effect of the intersection problem does not result in severe artefacts. It is however important to notice the effect when attempting the multiscale Hessian fracture filter on a different type of data. Also note that similar problems as for intersections may occur at fractures that terminate suddenly, as the tip of a fracture is also not a planar feature (also visible in Fig. 6). Additionally, very closely spaced fractures may interfere in the Hessian filtering step.

3.2. Other considerations and limitations

There are some other considerations before applying the presented filtering technique that cannot clearly be termed advantageous or disadvantageous.

It is important to notice the filtering works best on fractures cutting through an otherwise homogenous material. For example for μCT data, if the base material is very heterogeneous, with different phases showing up in a different intensity, artefacts may occur. This is because the Hessian filtering is sensitive to contrast, so also the boundary between two different phases with a high contrast is detected as a planar feature. Of course, the severity of such artefacts depends on the dataset (i.e. the contrast between different phases and the way the fractures appear), and in some cases a suitable preprocessing or calibration step afterwards may reduce the problems.

An important characteristic required for input data is spatial isotropy. The presented filtering technique does not support spatially anisotropic data. Every voxel should therefore be cubic, and the voxel dimensions should remain constant throughout a dataset. Although FeatureJ supports anisotropy, we choose not to support this in our approach, as this severely complicates or even prohibits the usage of the control lines. Proper assessment of the input data and proper conversions, if necessary, are thus important before starting the filtering. Other considerations (including data types, possible downsampling and the choice of the ROI) are explained in the Supplementary material.

3.3. Orientation information from the Hessian matrix

This paper focusses on the enhancement of fractures or other planar features, using intensities only, by using the Hessian matrix. Hence, only the information on magnitudes (the eigenvalues) has been used so far. We do want to point out however that also orientations can be extracted from the Hessian matrix, by using its eigenvectors. One can take the eigenvector orientation of the largest magnitude eigenvalue, for every voxel that is considered to be part of a planar feature by the filtering process. This eigenvector represents the normal direction to these planar features. These orientations can then for example be displayed and analysed using a stereographic projection. Tests where this technique for orientation analysis was used work well, but as FeatureJ does not natively output the full Hessian matrix, and additional processing is required, it is beyond the scope of this paper to elaborate on the orientation aspect here.

4. Conclusions

Narrow fractures in 3D computed tomography data can be problematic to extract, whereas this extraction is often necessary for future processing of the data. In this paper the multiscale Hessian fracture filtering approach is presented, providing an efficient way of extracting such narrow fractures from a dataset. Both the fuzzy output of the filtering approach, suitable for visualisation and porosity determination, and the fully binarised output display the good performance of this technique. This is especially apparent when comparing to direct global threshold

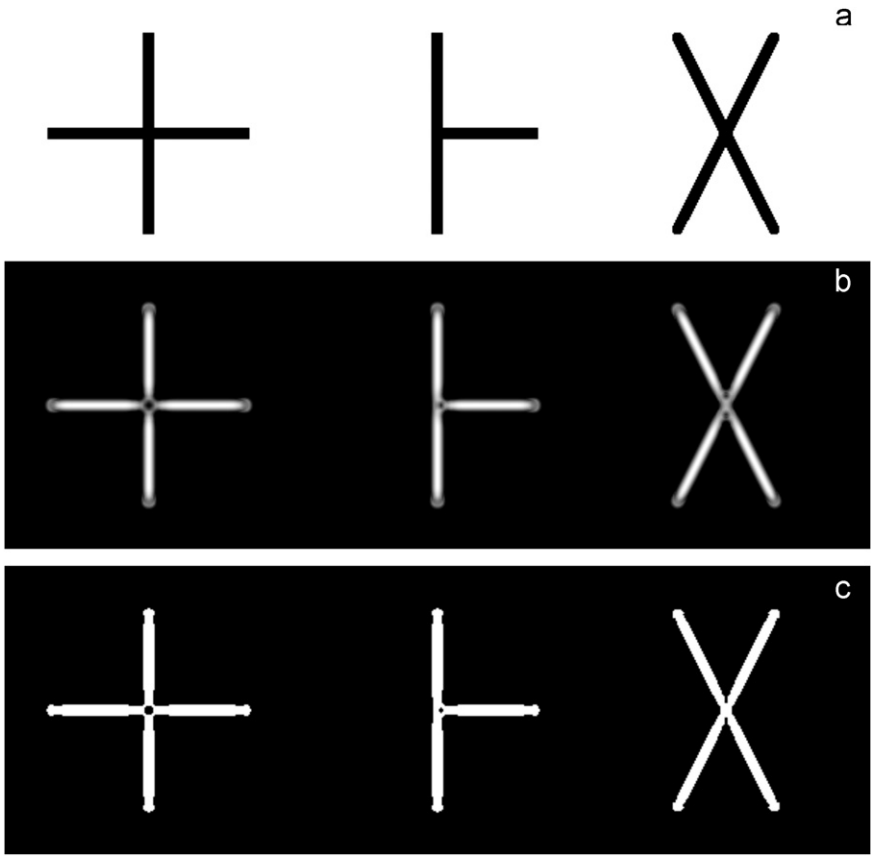

Fig. 6. Results of the multiscale Hessian fracture filtering for synthetic data with intersections and line terminations. Contrast and brightness settings have been altered for display. (a) Input data, the width of the features is originally 8 pixels. (b) Multiscale Hessian fracture filtering output (combination of Gaussian scales 2, 3 and 4). (c) Threshold of (b). It is clear from these figures, line terminations and intersections can pose problems for the filtering, as they do not represent planar data. Line terminations are represented in a bulging way, and line intersections become rounded with little response at the intersection core. The worst results for the intersections are shown for 90° angle full intersections (left), whereas the effects become less for terminating intersections (middle) and lower angle intersections (right). These examples can be seen as worst-case scenarios for synthetic data. See the main text for discussion why the influence of these intersections is often less problematic in real data.

binarisation on a dataset, but also for other non-structure-based segmentation techniques. The multiscale Hessian fracture filtering approach and the sequential postprocessing steps produce a result with little noise, in which even very narrow fractures are clearly emphasised. Furthermore, the multiscale Hessian fracture filtering technique is not hampered by most greyscale variations throughout datasets. The technique has some drawbacks, mainly inherent to using the Hessian matrix. The most important are the requirement of user input, and possible problems at fracture intersections and terminations. In our data examples, these effects however do not seem to introduce large artefacts.

The multiscale Hessian fracture filtering technique is implemented in the public domain software ImageJ, and provided as Supplementary material to this paper. The way of implementation ensures the data is processed in 3D, without user intervention after the initial setup, and without requiring extensive computational power (i.e. possible on a modern desktop PC). These properties are a clear advantage of the approach compared to other approaches in literature. The application of the technique has been shown on our μCT data of narrow fractures in reservoir dolomites. Besides the usage on these datasets, the general implementation of the technique provides a great potential for its usage on different 3D datasets, even when very different goals might be aspired.

Acknowledgements

The authors would like to thank OMV Exploration & Production for the funding of the research presented in this paper. Additional thanks goes to Christian Gusenbauer (University of Applied Sciences, Wels, Austria) for performing the μCT scans and reconstructions. The reviews of this paper, especially by Richard Ketcham, were very helpful to improve the message presented here. Editor-in-Chief Jef Caers and the other editors of Computers and Geosciences are thanked for their work for getting this paper published. Finally, the authors would like to acknowledge the developers of the used software FeatureJ (Erik Meijering), ImageJ (Wayne Rasband) and FIJI, for making the efforts of their work publicly available.

Appendix A. Supporting information

Supplementary information associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.cageo.2013.03.006. The MHSFF code (as well as possible future updates) is also available at https://github.com/cageo/Voorn-2013.

References

Christe, P.G., 2009. Geological Characterization of Cataclastic Rock Samples Using Medical X-ray Computerized Tomography: Towards a Better Geotechnical Description. Ph.D. Dissertation. Faculté Environnement Naturel, Architectural et Construit. École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, 338 pp.

Cnudde, V., Masschaele, B., Dierick, M., Vlassenbroeck, J., Hoorebeke, L., Jacobs, P., 2006. Recent progress in X-ray CT as a geosciences tool. Applied Geochemistry 21 (5), 826–832, http://dx.doi.org/10.1016/j.apgeochem.2006.02.010.

Descoteaux, M., Audette, M., Chinzei, K., Siddiqi, K., 2005. Bone enhancement filtering: application to sinus bone segmentation and simulation of pituitary surgery. In: Duncan, J.S., Gerig, G. (Eds.), Proceedings of Medical Image Computing and Computer-Assisted Intervention, Palm Springs, California, USA, MICCAI, 2005. Lecture Notes in Computer Science 3749, pp. 9–16.

Ehrig, K., Goebbels, J., Meinel, D., Paetsch, O., Prohaska, S., Zobel, V., 2011. Comparison of Crack Detection Methods for Analyzing Damage Processes in Concrete with Computed Tomography. In: Proceedings of International Symposium on Digital Industrial Radiology and Computed Tomography, DIR, Berlin, Germany, 2011, 8 pp.

Elliot, T.R., Heck, R.J., 2007. A comparison of 2D vs. 3D thresholding of X-ray CT imagery. Canadian Journal of Soil Science 87 (4), 405–412, http://dx.doi.org/10.4141/CJSS06017.

Fornaro, J., Székely, G., Harders, M., 2010. Semi-automatic Segmentation of Fractured Pelvic Bones for Surgical Planning. In: Bello, F., Cotin, S. (Eds.), Proceedings of the 5th International Symposium on Biomedical Simulation, ISBMS, Phoenix, Arizona, USA, 2010. Lecture Notes in Computer Science 5958, pp. 82–89.

Frangi, A.F., Niessen, W.J., Vincken, K.L., Viergever, M.A., 1998. Multiscale Vessel Enhancement Filtering. In: Wells, W.M., Colchester, A., Delp, S.L. (Eds.), Proceedings Medical Image Computing and Computer-Assisted Intervention, MICCAI, Cambridge, Massachusetts, USA, 1998. Lecture Notes in Computer Science 1496, pp. 130–137.

Iassonov, P., Gebrenegus, T., Tuller, M., 2009. Segmentation of X-ray computed tomography images of porous materials: a crucial step for characterization and quantitative analysis of pore structures. Water Resources Research 45 (W09415), 12, http://dx.doi.org/10.1029/2009WR008087.

Iassonov, P., Tuller, M., 2010. Application of segmentation for correction of intensity bias in X-Ray computed tomography images. Vadose Zone Journal 9 (1), 187–191, http://dx.doi.org/10.2136/vzj2009.0042.

Karpyn, Z.T., Grader, A.S., Halleck, P.M., 2007. Visualization of fluid occupancy in a rough fracture using micro-tomography. Journal of Colloid and Interface Science 307 (1), 181–187, http://dx.doi.org/10.1016/j.jcis.2006.10.082.

Keller, A., 1998. High resolution, non-destructive measurement and characterization of fracture apertures. International Journal of Rock Mechanics and Mining Sciences 35 (8), 1037–1050, http://dx.doi.org/10.1016/S0148-9062(98)00164-8.

Ketcham, R., Slottke, D.T., Sharp, J.M.J., 2010. Three-dimensional measurement of fractures in heterogeneous materials using high-resolution X-ray computed tomography. Geosphere 6 (5), 499–514, http://dx.doi.org/10.1130/ges00552.1.

Ketcham, R.A., Carlson, W.D., 2001. Acquisition, optimization and interpretation of X-ray computed tomographic imagery: applications to the geosciences. Computers and Geosciences 27 (4), 381–400, http://dx.doi.org/10.1016/S0098-3004(00)00116-3.

Landis, E.N., Nagy, E.N., Keane, D.T., 2003. Microstructure and fracture in three dimensions. Engineering Fracture Mechanics 70 (7–8), 911–925, http://dx.doi.org/10.1016/S0013-7944(02)00157-1.

Lindeberg, T., 1998. Edge detection and ridge detection with automatic scale selection. International Journal of Computer Vision 30 (2), 117–154, http://dx.doi.org/10.1023/A:1008097225773.

Lorenz, C., Carlsen, I.-C., Buzug, T.M., Fassnacht, C., Weese, J., 1997. Multi-scale Line Segmentation with Automatic Estimation of Width, Contrast and Tangential Direction in 2D and 3D Medical Images. In: Troccaz, J., Grimson, E., Mösges, R. (Eds.), Proceedings First Joint Conference, Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery, CVRMed-MRCAS, Grenoble, France, 1997. Lecture Notes in Computer Science 1205, pp. 233–242.

Mees, F., Swennen, R., Van Geet, M., Jacobs, P. (Eds.), 2003. Special Publications, vol. 215. Geological Society, London, ISBN: 1862391394 250 pp..

Meijering, E.H.W., 2010. FeatureJ 1.6.0, Biomedical Imaging Group Rotterdam, Erasmus MC, University Medical Center Rotterdam, The Netherlands, 2002–2010. ⟨http://www.imagescience.org/meijering/software/featurej/⟩ (accessed 13.12.12).

Porter, M.L., Wildenschild, D., 2010. Image analysis algorithms for estimating porous media multiphase flow variables from computed microtomography data: a validation study. Computational Geosciences 14 (1), 15–30, http://dx.doi.org/10.1007/s10596-009-9130-5.

Rasband, W.S., 2012. Image J, U.S. National Institutes of Health, Bethesda, Maryland, USA, 1997–2012. ⟨http://imagej.nih.gov/ij/⟩ (accessed 13.12.12).

Sato, Y., Nakajima, S., Atsumi, H., Koller, T., Gerig, G., Yoshida, S., Kikinis, R., 1997. 3D Multi-scale Line Filter for Segmentation and Visualization of Curvilinear Structures in Medical Images. In: Troccaz, J., Grimson, E., Mösges, R. (Eds.), Proceedings of First Joint Conference, Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery, CVRMed-MRCAS, Grenoble, France, 1997. Lecture Notes in Computer Science 1205, pp. 213–222.

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.Y., White, D.J., Hartenstein, V., Eliceiri, K., Tomancak, P., Cardona, A., 2012. Fiji: an open-source platform for biological-image analysis. Nature Methods 9 (7), 676–682, http://dx.doi.org/10.1038/nmeth.2019.

Sellers, E., Vervoort, A., Van Cleynenbreugel, J., 2003. Three-dimensional visualization of fractures in rock test samples, simulating deep level mining excavations, using X-ray computed tomography. In: Mees, F., Swennen, R., Van Geet, M., Jacobs, P. (Eds.), Applications of X-ray Computed Tomography in the Geosciences, Special Publications (215). Geological Society, Londonhttp://dx.doi.org/10.1144/GSL.SP.2003.215.01.07, pp. 69–80.

Sezgin, M., Sankur, B., 2004. Survey over image thresholding techniques and quantitative performance evaluation. Journal of Electronic Imaging 13 (1), 146–165, http://dx.doi.org/10.1117/1.1631315.

Stock, S.R., 1999. X-ray microtomography of materials. International Materials Reviews 44 (4), 141–164, http://dx.doi.org/10.1179/095066099101528261.

Stoessel, R., Wirjadi, O., Godehardt, M., Schlachter, A.-L., Liebscher, A., 2012. Analysis of inner fracture surfaces in CFRP based on μ-CT image data. In: Proceedings Conference on Industrial Computed Tomography, ICT, Wels, Austria, pp. 33–40.

The Mathworks, Inc., 2011. MATLAB® R2011a, Natick, Massachusetts, USA. ⟨http://www.mathworks.com/products/matlab/⟩ (accessed 13.12.12).

Van Geet, M., Swennen, R., 2001. Quantitative 3D-fracture analysis by means of microfocus X-ray computer tomography (μCT): an example from coal. Geophysical Research Letters 28 (17), 3333–3336, http://dx.doi.org/10.1029/2001GL013247.

Van Geet, M., Swennen, R., Wevers, M., 2000. Quantitative analysis of reservoir rocks by microfocus X-ray computerised tomography. Sedimentary Geology 132 (1–2), 25–36, http://dx.doi.org/10.1016/S0037-0738(99)00127-X.

Visualization Sciences Group (VSG), 2012. Amira® 5, Merignac, France. ⟨http://www.amira.com/⟩ (accessed 13.12.12).