Automatic crack detection on concrete floor images

Simler, Christophe, Trostmann, Erik, Berndt, Dirk

Christophe Simler, Erik Trostmann, Dirk Berndt, "Automatic crack detection on concrete floor images," Proc. SPIE 11144, Photonics and Education in Measurement Science 2019, 111440T (17 September 2019); doi: 10.1117/12.2531951

SPIE.

Event: Joint TC1 - TC2 International Symposium on Photonics and Education in Measurement Science 2019, 2019, Jena, Germany

Automatic crack detection on concrete floor images

Christophe Simler, Erik Trostmann, Dirk Berndt
Fraunhofer Institute for Factory Operation and Automation IFF, Sandtorstraße 22, 39106 Magdeburg, Germany

ABSTRACT

This paper presents an algorithm detecting automatically a wide variety of cracks on monochrome images of concrete floors. It is part of a vision system, generating a crack map to support the condition monitoring for buildings. The suggested method uses successively radiometric, geometric and contextual information. An automatic supervised adaptive intensity-threshold method handles radiometry information. A threshold-based method was chosen because it can separate even thin cracks from the background. In order to cope with different image cluster intensities, first a clustering algorithm separates regions of different intensities, and second the threshold is adaptive at the cluster level; being a continuous function of the cluster intensity. At some points, function values were learned via supervised classification. Then, we performed an interpolation between these points in order to get a threshold whatever the cluster intensity (continuity). At this step, we have a thresholded image. However, due to texture, non-crack dark defects and dirtiness we have false positives. In order to overcome this problem, connected pixels are grouped into regions, and after discarding small regions, a size-dependent geometrical shape filter is suggested. The shape of a crack region depends on its area, and this relation was empirically learned. A region is retained if its shape features are above some area depending thresholds. However, mainly due to texture, we have still false positives because some non-crack entities have radiometry and geometry of cracks. Fortunately, they are often small isolated regions and are discarded via an isolation filter. Tests performed on many images show very encouraging results.

Keywords: Crack detection, inspection, adaptive threshold, clustering, supervised classification, learning, feature extraction, expert rules

1. INTRODUCTION

In building monitoring concrete floor inspection, generating a crack map enables assessing important information about the surface health. In addition, detecting cracks early minimizes maintenance-renovation efforts and costs. Currently, crack detection is mainly performed by human experts. It is a tiring, time consuming, costly and above all subjective task. An improvement is to use scanning vision systems and possible basic image processing algorithms highlighting at least some cracks. In this case, experts just have to visualize the image sequences. However, it is still suggestive. In order to overcome these problems, many efforts are currently done to automatize the inspection. Cracks generally present radiometric and geometric features such as darker than the background and elongated. These are the reasons why scanning vision systems based on a monochrome camera, image processing and recognition are natural tools for this task. These tools are used in this work. However, it is still a current challenge to detect automatically (and fast) a wide variety of cracks in a wide variety of concrete floors. This article proposes a crack detector tending to perform this goal. Part 2 presents the main crack detection approaches, their limitations and at the end, the position of our work with respect to them. Part 3 describes our system and crack detection algorithm. The results are in part 4, and part 5 is the conclusion.

2. RELATED WORK

For the last couple of decades, a large variety of crack detection algorithms have been developed. Overviews can be found in [1, 2]. Some procedures provide only the information if there are cracks or not in the image or image patches, without localizing the crack [7, 21], and in contrast some others provide a crack map (at the pixel level). Both approaches are considered here, but we remind that our algorithm, explained in part 3 of this paper, generates a crack map. Having a crack map, connected crack pixels can be grouped and geometrical features of each crack such as length, width, gravity center and orientation can be extracted. This enables, for example, visualizing only cracks belonging to some geometric categories (geometric filter). However, this article is only about crack detection; crack feature extraction

Photonics and Education in Measurement Science 2019, edited by Maik Rosenberger, Paul-Gerald Dittrich, Bernhard Zagar, Proc. of SPIE Vol. 11144, 111440T
 · doi: 10.1117/12.2531951

and exploitation are the next step and are beyond the scope of this paper. Crack detection algorithms are generally based on threshold, edge detection, percolation, texture/shape, mathematic morphology, pattern recognition and more recently deep learning.

With a threshold-based method, a first strategy consists in performing an initial pre-processing against non-uniform lighting (and possibly against noise and texture) [4, 5, 6, 8], and then applying a global threshold over the corrected image. Unsupervised approaches provide a threshold depending on the image (flexible), the advantage is that there is no need to control the lighting. In this category there is the famous Otsu method [3, 4, 5, 6], the expectation maximization (EM) clustering, and empirical methods computing a threshold with respect to image features, such as the average intensity and the minimal value [8]. However, the Otsu method can fail with cracks because often the histogram is rather unimodal due to the small proportion of crack pixels. Supervised approaches fixe the threshold via learning, but the lighting should be controlled. A second strategy with threshold-based method, robust with respect to mon-uniform lighting, is adaptive thresholding. The image is partitioned into cells having quasi-uniform illumination. In [9], the threshold of selected cells is function of the average intensity value. In [10], a probabilistic relaxation for coarse crack detection and noise elimination is performed to produce a binary image without any parameter to optimize. Then, an improved locally adaptive thresholding is performed via the Otsu method to close undesirable unconnected cracks and to extend cracks (fine detection). Results are better than with the global thresholding in terms of both sensitivity and specificity. [25] suggests an approach able to cope with the complex background texture, uneven illumination and non-uniform background of roads. Cracks are seen as inhomogeneities in the background texture. The image is divided into overlapping cells, and some co-occurrence matrices of each cell are computed. These cell texture features are concatenated with two shape descriptors. The shape descriptors are used as discriminative features against uneven illumination. They are the principal axis of the cell and the eccentricity on the non null terms in the co-occurrence matrix. The feature vector of the cell is classified via SVM. A pixel-wise classification using histogram information is applied to obtain the crack map for cells labeled as cracks. Generally the thresholded image contains cracks and false positives. Geometry can be used to overcome this problem. Morphological opening can possibly be first applied for regularization. Then connected pixels are grouped in regions and geometrical features feed a classifier aiming at separating cracks regions from the noise. In [8], these features are eccentricity, width and length. In [9] a line fitting is performed in each cell and radiometry is also used at this level for the classification of the cell via SVM. In [6], a morphological algebraic area opening operation filters the small connected components. Then, region features are the average intensity, the area and the standard deviation of shape distance histogram, and the ELM supervised classifier is used. In [25], small or block-like connected components are eliminated, and components with same orientation and space adjacently are merged together. Only real linear cracks remain.

With edge-based methods, the advantage over threshold approaches is that edges are more robust with respect to non-uniform illumination in the image, change of illumination conditions and shading. However, noise filtering should be applied in a pre-processing step (scaling), and obtained segments are often not continuous. Among the most famous edge detectors we have Canny, Laplacian of Gaussian (LoG) with zero crossing [11, 12] and the Fast Haar Wavelet Transform (FHT, it is a multi-scale edge detector). In [13], it is shown that FHT provides better results for crack detection than Canny. Percolation-based methods are homogenous to edge-based methods, but consider crack connectivity among neighboring pixels in order to be less affected by noise [15]. For a considered pixel, at each iteration a threshold is updated and a "percolated region" is growing, up to convergence. At the end, the pixel is classified as crack pixel if the circularity of the percolated region is below a threshold. Thus, geometric information is already used during a pixel-wise classification. The problem is that the basic method is time consuming and has four parameters. Faster versions were designed in [16, 17]. Like with the previous threshold methods, pixel grouping can be done to form regions in the binary image (for each scale). Now the regions are the isolated curves. Geometric and radiometric features can be extracted to feed a classifier (pattern recognition). In [11], a semi-automatic graph theoretic segmentation is applied to the ravine network, then regions are filtered according to their geometry. If for the percolation-based methods it is not so clear because more tests should be done, generally the edge-based methods are not the most successful to detect cracks [9, 14].

The neighbor of the pixel can be used to form a texture/shape pixel pattern. In [18], histograms of oriented gradients (HoG) are used to build the pixel pattern, which is classified via SVM.

Deep learning, and in the case of image data, deep convolutional neural networks (DCNNs) tends to replace all the previous technics. DCNNs are neural network classifiers, but normally having as an input just the raw image. Therefore, there is no image processing and tedious feature extraction any more. During the learning of a DCNN, the kernel weights of many convolution matrix (neurons, parameters) are iteratively updated to automatically learn the convolutional

invariant features of the input image. A drawback is that the training set should contain many examples. The loss function is generally a regularized average cross-entropy. Its minimization, performed via stochastic gradient descent can be time consuming even with GPUs. DCNNs are intrinsically image or image patch classifiers. This is why in the literature, in most of the works, the input image is scanned with a patch, and for each patch the DCNN predicts if it contains cracks or not (patch classification level) [21]. In [19] patches containing relatively clear cracks are detected, but a large number of false positives also and the method is only compared with basic algorithms. [20] has also the problem of false positives at the CNN output, but discard them using multiple frames in post-processing (context of video sequence). [21] discusses about fast but sub-optimal learning methods. The structure of DCNNs can be extended in order to provide a prediction in the form of a probabilistic crack prediction map (pixel classification level, and all the pixels are simultaneously classified because of the correlation between pixels), it is also called in the literature "semantic segmentation image". In [22], a fully convolutional network model (FCN) is suggested to detect cracks at the pixel level. However, the detected cracks showed in this article are large and dark. Whatever the method, they would have been detected anyway. In [23], another architecture for pixel level using DCNN encoder and decoder to produce and merge feature maps from different scales is suggested: the DeepCrack. It is an improvement of the SegNet architecture, and results are encouraging. However, the cross entropy loss function in now pixel-wise, and thus the numerous ground truth crack class maps preparation for the training set require a lot of amount of work.

Among the above existing methods, the threshold-based ones are particularly attractive for the detection of a wide variety of cracks, notably the thin ones. This is the reason why we have chosen such an approach in this article. Indeed, edge-based methods such as Canny require noise filtering and this operation generally damage too much the fine cracks, without entirely solving the problem of false positives. In addition, approaches consisting of a direct segmentation of the input image generally fail (mean shift for segmentation, watershed …), because thin cracks are often not segmented, and false positives can be difficult to discard. Concerning recent approaches, the classical DCNNs, detecting crack presence in image patches on an image grid, are discarded because our aim is to generate a crack map (at the pixel level). The DeepCrack architecture overcomes this limitation, but the learning is time consuming and above all the preparation of a reliable training set is challenging.

3. SYSTEM AND SUGGESTED CRACK DETECTION ALGORITHM

Our mobile platform can be seen on figure 1. During the operational scenario, the system is moved manually around a whole parking, taking an image sequence covering the planar parking ground. We use a standard monochrome industrial camera (4112x3008 pixels), oriented toward the ground. Successive images have no significant overlap, it is not a video sequence. In order to build a global crack map in the absolute parking frame, the mobile platform localizes itself. It uses the Monte Carlo Localization (particle filter) algorithm, with laser scanner (see figure 1) and odometry; knowing a map of the environment. Due to code efficiency optimization via GPU, the computational time to obtain the crack map of one image is about 2 s (almost real time). A spatial resolution of 0.12 mm/pixel enables to detect cracks larger than 0.25 mm. In fact, cracks larger than 1.5 or 2 pixels can be detected. The resolution is a parameter of the program. The algorithm is robust with respect to resolution change, and no re-training is necessary. Controlled lighting and shading correction enable having a uniform illumination on our images. Shading correction is implicit in the following.

Figure 1: The mobile platform with its operating panel. The laser scanner, used to localize the system is at the top of the device. The camera, used for crack detection is not visible because it is under the system, acquiring the parking ground.

Our full automatic crack detection algorithm respectively uses radiometry, geometry and context information to build the crack map. The radiometric pixel-wise 2-class classification is threshold-based. Adaptive threshold at the cell or pixel level is not needed because we have a uniform lighting. However, the threshold cannot be global because the background of our images can have several clusters (see the grey and white clusters of figure 2). Therefore, the threshold should be adaptive at the cluster level. The clusters are first isolated by the mean shift clustering algorithm (image partition into clusters) [24]. Then, the challenge is to compute for each cluster an intensity threshold separating the cracks from the cluster. Generally, the Otsu threshold fails because crack pixels are too minority to form with the cluster a real bimodal histogram. In addition, it is not straightforward to establish an empiric reliable closed form formula computing the threshold from the cluster intensity. Because of these difficulties, the threshold is determined with a simple supervised classification approach. We collected images from a wide variety of parking. In these images, we selected a set of clusters having a large difference in intensity/texture from each other, and containing each a wide variety of cracks. For each cluster, we learned a threshold from a two class training set containing cluster and crack pixel intensities. At this point, a threshold separating the cracks is associated to each cluster intensity of our set. In order to associate a threshold whatever the cluster intensity, a piecewise linear function is fitted to these points. We make the assumption, that the relation between threshold and cluster intensity can be modelled with a single curve; and independently of the textures of the different parking surfaces. In this article, the cluster intensity is the label intensity value provided by the mean shift algorithm after convergence, it is similar to the cluster median intensity.

At this step, the radiometric information has been entirely exploited, and as result we have a binary threshold image (automatically set). In the example of figure 2, the cracks are well detected on the different clusters. There is no significant problem with false negative. However, due to texture and non-crack dark entities (other defects, dirtiness) on the image, many detected pixels are no crack. We are in the presence of false positives.

In order to overcome this problem, first an area filter eliminates the numerous very small (area less than 200 pixels) regions, due to the texture, and the possible huge regions of the threshold image. Then, a morphological opening fills the small holes in the connected components. After this cleaning of the binary threshold image, the connected pixels are grouped into regions (object approach) and a geometrical shape filter related to the area is suggested. In this work, shape features are eccentricity, compactness and concavity (see table 1). The shape of a crack region depends on its area, and this relation was empirically learned. Knowing the area of a region, the shape filter classify it as crack if its shape feature values are upper than some thresholds (see figure 3).

- The first main insight is that small cracks generally have large eccentricity. This constraint is progressively released when the region becomes larger.
- The second main insight is that large cracks generally have large compactness. This means that the single shape feature is that it is not compact. This constraint is progressively released when the region becomes smaller.
- Regions not too small having huge compactness are automatically retained in order to keep the stars. A star occurs when from a point, a crack goes in more than two directions. The image on figure 2 contains a star at the top.
- The concavity is used to discard regions with chaotic borders. It is large for all the cracks, especially for the small ones.

Table 1. Shape features. In green: shape features shared by some crack categories. Small cracks have large eccentricity (line 1). Large cracks have large compactness (line 2). All the cracks have large concavity (line 3).

Eccentricity [0; 1]	Low: 0	Low	Large -> 1 (small cracks)
Compactness [4π; ∞[	Low: 4π	Large (large cracks)	Large (large cracks)
Concavity [0; 1]	Large:1 (all the cracks)	Low -> 0	Large (all the cracks)

Despite the geometric filter, we have still a problem with false positives. Mainly due to the texture (granularity) on our images with some parking, it happens that some non-crack entities have intensity and geometry of cracks. Fortunately, these false positives are often small isolated regions (area less than 700 pixels). In order to cope with this problem, an isolation filter discards them (use of contextual information). The previous area filter and the isolation filter are essential against the false positives at their respective scales (200 and 700 pixels).

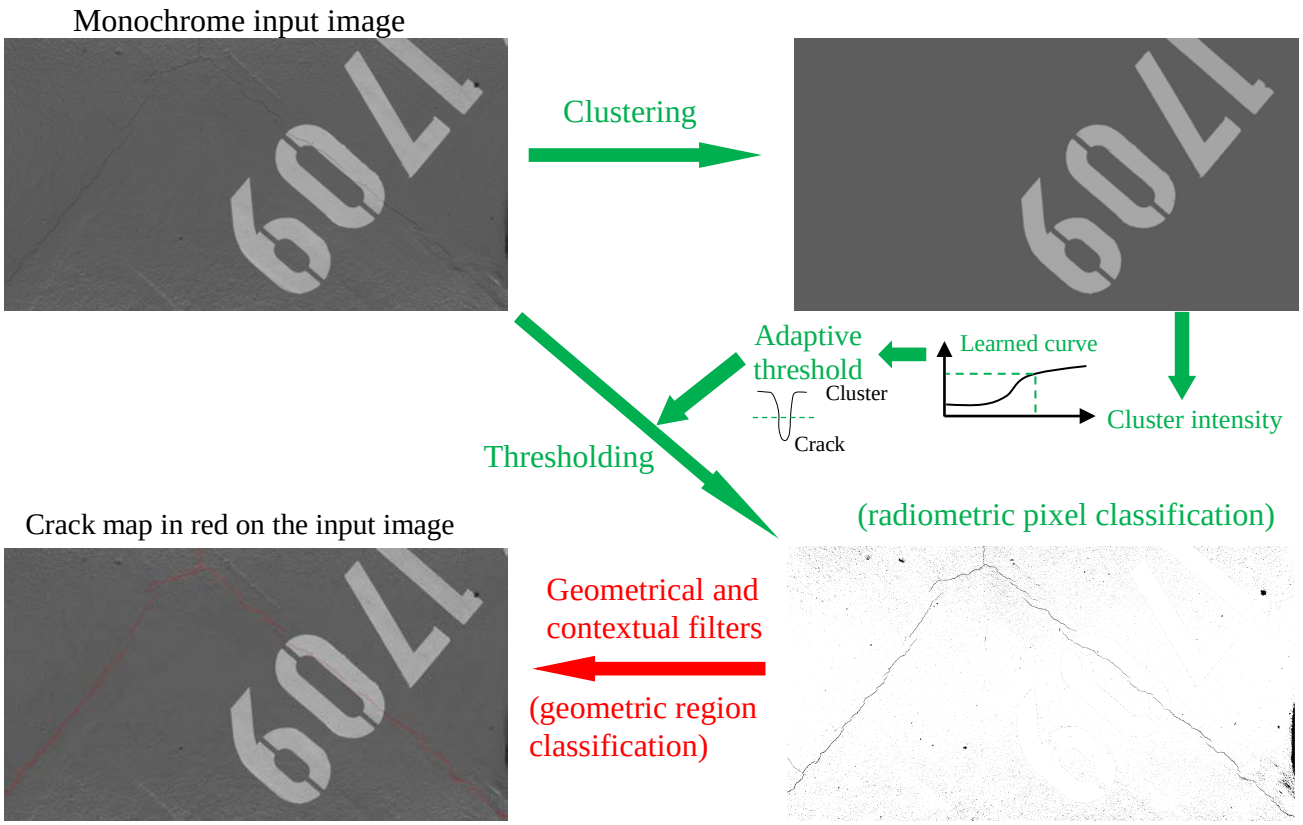


Figure 2: Flow chart of the crack detection algorithm. Comment of the result for this image: the crack is almost entirely detected in both clusters, and there is no false positive.

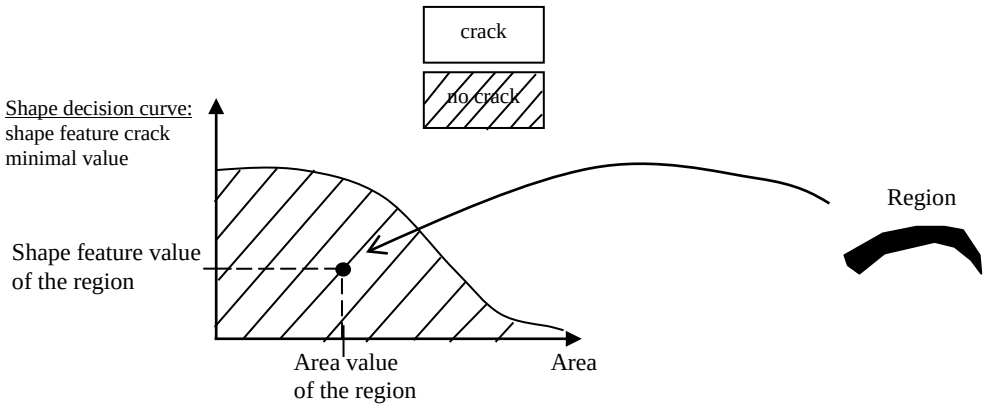


Figure 3: The size-dependent shape filter. In this example, the region is discarded. According to the classifier, its shape feature value is not high enough to be a crack.

However, by using the area and isolation filters in that way, there is a price to pay for such robustness against false positives. Let us consider thin cracks (about two pixel width). Unfortunately, there is an additional complication with these cracks because they often have natural discontinuities. Nevertheless, they are generally visible in the threshold image (with radiometry, they are detected). This is because they are often relatively dark, and the threshold was learned to separate them from the clusters. The problem is, because of the thinness and discontinuities of such cracks, it can happen that some regions have areas smaller than 200 pixels and are discarded by the area filter. This increases the discontinuities between remaining (larger) regions and thus their isolations. The regions having areas smaller than 700 pixels, and which are now isolated are discarded by the isolation filter. This mechanism can finally suppress significant parts of thin cracks. An illustration is shown in figure 4, and real examples are shown in the next part.

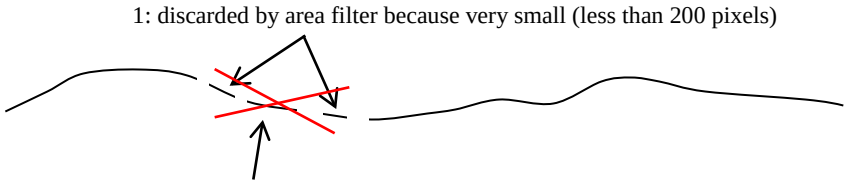
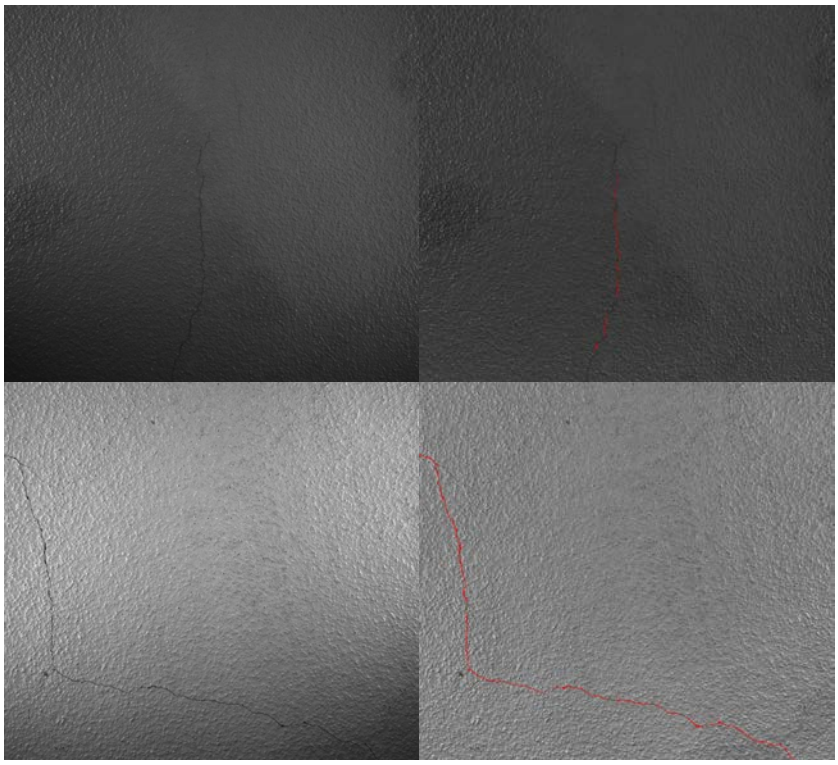


Figure 4: A thin crack with natural discontinuities, visible in the threshold image. In that case, the area and isolation filters can suppress significant parts of the cracks. It is the price to pay to limit the false positives due to texture.

4. RESULTS

The left column of figure 5 shows examples of input raw images (before shading correction) containing various kinds of cracks with different widths, on different concrete floors. Thin cracks are often naturally discontinuous. The image of a concrete surface can be dark, grey, clear, with texture (granularity), with important dirtiness, and with different clusters like in the previous example of figure 3. The right column shows the detected crack pixels in red (crack map), on the input image after shading correction.



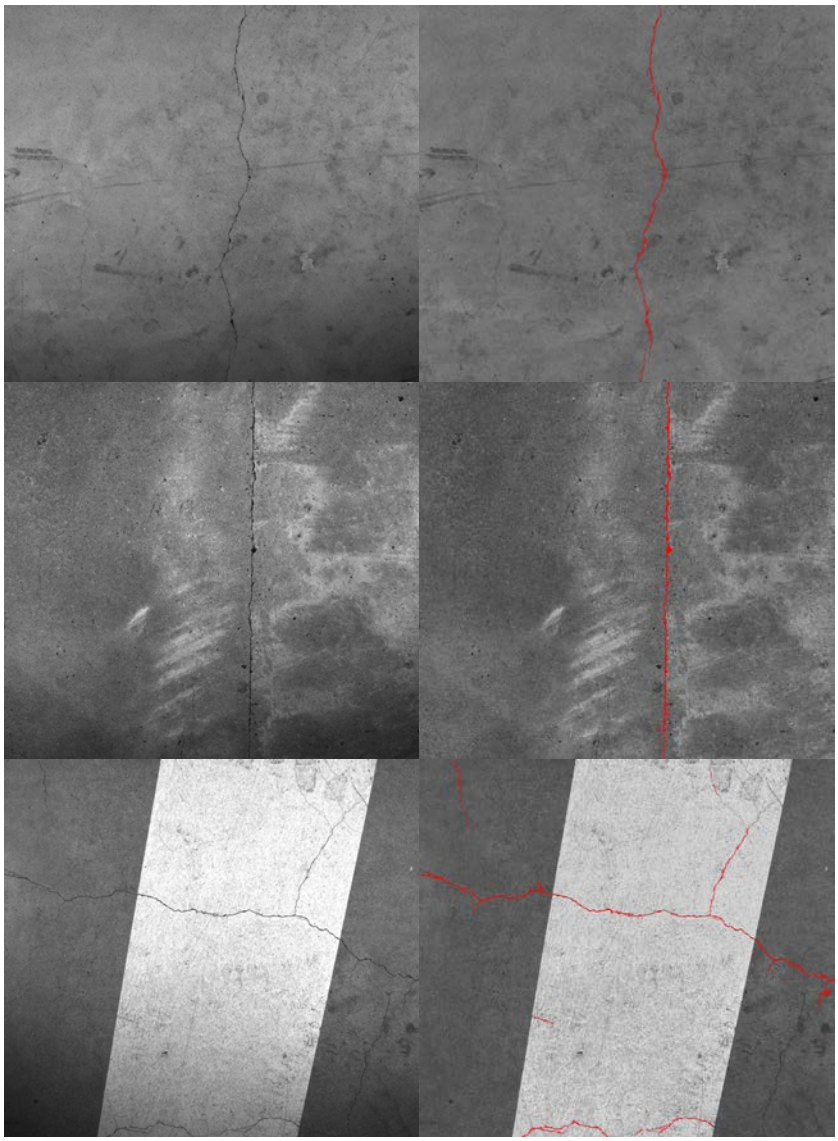


Figure 5: Left: input images containing various kinds of cracks with different widths, on different concrete floors. Right: crack map in red.

Table 2. Surface, crack features and detection for the images of figure 5.

Image of figure 5:	1	2	3	4	5
Surface	Dark, textured	Clear, textured	Clear, smooth, dirtiness	Clear, smooth, huge dirtiness	Grey and white clusters
Crack width [pixel]	<2	>2	> 1.5	>2	>1.25 and <2.5
False positive	No	No	No	No	No
Crack detection	Some parts not detected because thin and discontinuity	Complete (more than 95%)	Complete (more than 95%)	Complete (more than 95%)	Some parts not detected because thin and discontinuity

A large number of images were analyzed, enabling to establish the following comments:

Generally, the threshold-based method enables detecting with success cracks wider than 1.5 or 2 pixels (more than 95% of their pixels are detected). In other words, at this scale we have few problems with false negative. False positives, coming mostly from the texture, are seldom because of the size, size-dependent shape and isolation filters.

Cracks finer than 1.5 or 2 pixels are generally partially detected because of the presence of natural discontinuities. They are generally visible in the threshold image, but often parts are discarded (natural discontinuities are amplified) by the area and isolation filters (explanations at the end of part 3). When the width approaches 1 pixel, generally just a small proportion is detected (possibly 10%). However, although the detector cannot provide a suitable crack map, it offers almost 100% success when considering just crack presence detection ability in an image.

False positive: with some concrete surfaces, our images are textured (see the two first images of figure 5). The texture comes from the combination of the surface and the lighting. It can produce false positives, nevertheless with an area often lower than 700 pixels, and the area and isolation filters discard them in general. It can happen that two false positive regions coming from the texture, have areas between 200 and 700 pixels, and are close to each other. In this case, neither the area nor the isolation filter can eliminate them and we obtain a “double” false positive. Without such texture, constraints on these filters could be released and that would improve the detection of fine cracks.

False negative: texture can also possibly, but seldom, provide false negative, because it changes the crack shape (chaotic border). Dark dirtiness are generally discarded by the filters. However, they can create false negatives. If a crack is in contact with such dirtiness, the geometry of the union is generally discarded by the shape filter.

5. CONCLUSION AND PERSPECTIVES

The presented crack detector is able to detect a wide variety of cracks on different concrete floors. First, a threshold image is build, on which even thin cracks are generally visible. However, this image suffers from false positives. The combination of size, size-dependent shape and isolation filters enables generally to discard them. However, cracks finer than 1.5 or 2 pixels are generally also naturally discontinuous, and the size and isolation filters often partially discard them. Thus at this scale, we have a problem of false negative. A better detection of these fine cracks is possible by decreasing the filter selectivity, but we would have much more false positives due to the texture on some of our images. Texture results on the combination of the surface and the lighting, and it is not straightforward to find a lighting configuration providing un-textured images with some surfaces. It is also not easy to filter such texture in pre-processing, because this texture depends on the surface, is not constant over the image and any filter operation would damage the cracks.

In summary, cracks finer than 1.5 or 2 pixels are generally visible in the threshold image and are partially detected. Cracks wider than 1.5 or 2 pixels are generally detected with success, and we have no significant problem of false positives.

Perspectives:

Crack feature extraction: after the establishment of a crack map, the next step is to extract geometric crack features such as length, width, gravity center and orientation. These features can be obtained via moment computation; diagonalization of the covariance matrix of the crack. However, when a crack is discontinuous, or have unusual geometry, the problem can be difficult and we will investigate in these directions.

Comparisons with other methods: the suggested approach first thresholds the input image. Then, the connected pixels are grouped into regions (object approach) and our geometrical shape filter related to the area is applied. Instead of using such shape filter, an alternative pattern recognition strategy is to form a region pattern with geometric features such as area, eccentricity, compactness and concavity; and to classify it via SVM for example, before applying the final contextual filter. The area feature is essential, knowing that the shape of a crack is correlated with its area; radiometric features would be useless because radiometry was already fully used to threshold the input image (all our objects, cracks or not, are dark in the input image). Comparisons between our algorithm and this second threshold-based pattern recognition approach will be performed. It is also planned to compare our method with DCNNs performing semantic image segmentation, such as for example the DeepCrack architecture.

ACKNOWLEDGMENTS

The project was co-financed by the European Fund for Regional Development (EFRE) and the state of Sachsen-Anhalt (funding code: 1704/00036).

REFERENCES

[1] Koch, C., Georgieva, K., Kasireddy, V, Akinci, B. and Fieguth, P., "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," Advanced Engineering Informatics 29(2), 196–210 (2015).
[2] Mohan, A. and Pootbal, S., "Crack detection using image processing: A critical review and analysis," Alexandria Engineering Journal 57(2), 787-798 (2018).
[3] Otsu, N., "A threshold selection method from gray-level histograms," IEEE Trans. Sys., Man., Cyber. 9(1), 62–66 (1979).
[4] Oliveira, H. and Correia, P. L., "Automatic crack detection on road imagery using anisotropic diffusion and region linkage," 18th European Signal Processing Conference (EUSIPCO-2010), 274-278 (2010).
[5] Fujita, Y., Mitani, Y. and Hamamoto, Y., "A method for crack detection on a concrete structure," ICPR, (2006).
[6] Zhang, W., Zhang, Z., Qi, D. and Liu, Y., "Automatic crack detection and classification method for subway tunnel safety monitoring," Sensors 2014 14(10), 19307-19328 (2014).
[7] Abdel-Qader, I., Pashaie-Rad, S., Abudayyeh, O. and Yehia, S., "PCA-based algorithm for unsupervised bridge crack detection," Adv. Eng. Softw. 37 (12), 71–778 (2006).
[8] Ito, A., Aoki, Y. and Hashimoto, S., "Accurate extraction and measurement of fine cracks from concrete block surface image," IEEE IECON 02, 2202-2207 (2002).
[9] Prasanna, P., Dana, K., Gucunski, N. and Basily, B., "Computer vision based crack detection and analysis," Proc. SPIE 8345, Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems, (2012).
[10] Fujita, Y. and Hamamoto, Y., "A robust automatic crack detection method from noisy concrete surfaces," Machine Vision and Applications 22(2), 245–254 (2011).
[11] Yu, S. N., Jang, G. A. and Han, C. S., "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel," Automation in Construction 16(3), 255–261 (2007).
[12] Lim, S. R., La, H. L. and Sheng, W., "A robotic crack inspection and mapping system for bridge deck maintenance," IEEE Trans. On Automation Science And Engineering, 11(2), 367-378 (2014).
[13] Abdel-Qader, I., Abudayyeh, O. and Kelly, E. M., "Analysis of edge-detection techniques for crack identification in bridges," J. Comput. Civ. Eng., 17(4), 255-263 (2003).
[14] Hutchinson, T. C. and Chen, Z., "Improved image analysis for evaluating concrete damage," Journal of Computing in Civil Engineering 20(3), 210-216 (2006).
[15] Yamaguchi, T., Hashimoto, S., "Fast crack detection method for large-size concrete surface images using percolation-based image processing," Machine Vision and Applications 21(5), 797-809 (2010).
[16] Yamaguchi, T. and all., "Image-based crack detection for real concrete surfaces," Trans. On Electrical and Electronic Engineering IEEJ Trans 2008, 3(1), 128-135 (2008).
[17] Zhu, Z., German, S. and Brilakis, I., "Visual retrieval of concrete crack properties for automated post-earthquake structural safety evaluation," Automation in Construction 20(7), 874-883 (2011).
[18] Kapela, R. and all., "Asphalt surfaced pavement cracks detection based on histograms of oriented gradients," Proceedings of the 22nd International Conference "Mixed Design of Integrated Circuits and Systems", 579-584 (2015).
[19] Cha, Y. J. and Choi, W., "Deep learning-based crack damage detection using convolutional neural networks," Computer-Aided Civil and Infrastructure Engineering 32(5), 361-378 (2017).
[20] Chen, F., C. and Jahanshahi M. R., "NB-CNN: Deep learning-based crack detection using convolutional neural network and naive Bayes data fusion," IEEE Trans. On Industrial Electronics 65(5), 4392-4400 (2018).
[21] Dorafshan, S., Thomas, R. J. and Maguire M., "Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete," Construction and Building Materials journal 186(20), 1031-1045 (2018).

[22] Li, S., Zhao, X. and Zhou, G., “Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network,” Comput Aided Civ Inf., 1-19 (2019).
[23] Zou, Q., Zhang, Z., Li, Q., Qi, X., Wang, Q. and Wang, S., “DeepCrack: Learning hierarchical convolutional features for crack detection,” IEEE Trans. On Image Processing 28(3), 1498-1512 (2019).
[24] Comaniciu, D. and Meer, P., “ Mean Shift: A robust approach toward feature space analysis,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(5), 603-619 (2002).
[25] Hu, Y., Zhao, Chun-xia. and Wang, H-N., “Automatic pavement crack detection using texture and shape descriptors,” IETE Technical Review, 27(5), 398-405 (2010).