Concrete Bridge Crack Image Classification Using Histograms of Oriented Gradients, Uniform Local Binary Patterns, and Kernel Principal Component Analysis

Hajar Zoubir 1, Mustapha Rguig 1, Mohamed El Aroussi 2, Abdellah Chehri 3,* and Rachid Saadane 2

1 ERIC-(LAGCET), Hassan School of Public Works, Casablanca BP 8108, Morocco
2 SIRC-(LaGeS), Hassan School of Public Works, Casablanca BP 8108, Morocco
3 Department of Mathematics and Computer Science, Royal Military College of Canada, Kingston, ON K7K 7B4, Canada
* Correspondence: chehri@rmc.ca

Abstract: Bridges deteriorate over time, which requires the continuous monitoring of their condition. There are many digital technologies for inspecting and monitoring bridges in real-time. In this context, computer vision has extensively studied cracks to automate their identification in concrete surfaces, overcoming the conventional manual methods that rely on human judgment. The general framework of vision-based techniques consists of feature extraction using different filters and descriptors and classifier training to perform the classification task. However, training can be time-consuming and computationally expensive, depending on the dimension of the features. To address this limitation, dimensionality reduction techniques are applied to extracted features, and a new feature subspace is generated. This work used histograms of oriented gradients (HOGs) and uniform local binary patterns (ULBPs) to extract features from a dataset containing over 3000 uncracked and cracked images covering different patterns of cracks and concrete surface representations. Nonlinear dimensionality reduction was performed using kernel principal component analysis (KPCA), and three machine learning classifiers were implemented to conduct the classification. The experimental results show that the classification scheme based on the support-vector machine (SVM) model and feature-level fusion of the HOG and ULBP features after KPCA application provided the best results as an accuracy of 99.26% was achieved by the proposed classification framework.

Keywords: concrete bridge cracks; feature extraction; dimensionality reduction; histograms of oriented gradients; uniform local binary patterns; kernel principal component analysis

1. Introduction

Cracks in concrete bridges are hazardous to their durability and can potentially lead to their structural deficiency. These defects are largely encountered in bridges while performing visual inspections, and inspectors need to carefully identify them and document their different properties to assess the condition of the examined bridge.

Automating the detection of this type of defect has been an active research area in the computer vision field in an attempt to overcome limitations related to traditional manual methods that essentially rely on human judgment.

Generally, computer-vision-based detection methods include extracting features from a set of images using edge detection filters (e.g., Sobel, Canny, and Prewitt) [1,2] and feature descriptors (e.g., histogram of oriented gradients, scale-invariant feature transform, and speeded up robust features) [3,4].

The extracted features are fed to a classifier (e.g., support-vector machine [5], random forests [6] and AdaBoost [7]). Another framework where features are automatically learned from a training set of images has been extensively explored leveraging convolutional neural networks, transfer learning, and several public concrete crack datasets [8–12].
In the context of handcrafted features-based frameworks, Choudhary and Dey [13] used a Sobel edge detector to extract features from digital images of concrete surfaces and trained neural network and fuzzy models to detect concrete cracks.

Chen et al. proposed a detection method for highway pavement damage [14]. The authors used a grayscale-weighted histogram of oriented gradient feature patterns and a convolutional neural network classifier.

Jin et al. [15] used the histogram of oriented gradients and the watershed algorithm to detect different pavement cracks and identify their severity.

Another example was proposed by Zalama et al. [16] The authors developed a methodology utilizing Gabor filters, several binary classifiers, and an AdaBoost algorithm to detect longitudinal and transverse pavement cracks.

Chen et al. [17] presented a texture-based video-processing approach using local binary patterns (LBP), support-vector machine (SVM), and Bayesian decision theory to automate crack detection on metallic surfaces. Finally, Hu et al. [18] proposed a pavement crack detection method based on texture analysis using the gray-level co-occurrence matrix, translation invariant shape descriptors, and an SVM classifier.

Depending on the dimension of the extracted features, classifier training can be time-consuming and computationally expensive. This limitation can be addressed by applying dimensionality reduction techniques.

For example, principal component analysis, linear discriminant analysis, or isometric mapping can be used as a processing step in the general classification framework.

In this sense, Abdelmawla et al. [19] applied image processing techniques, principal component analysis (PCA), and the k-means algorithm to detect and classify different types of cracks in pavement surface images. In addition, Abdel-Qader et al. [20] proposed an algorithm based on PCA, linear features, and local information to detect cracks in concrete bridge images.

Kumar et al. [21] used PCA and a modified LeNet-5 model to detect cracks in roads and bridges working with three different crack datasets. In addition, Endri et al. [22] used PCA as a preprocessing step to extract features from a pavement crack dataset containing 400 images.

Chen et al. [23] extracted features from frames of road videos using LBP, reduced the dimension of the LBP feature space by PCA, and trained an SVM classifier to detect different types of pavement cracks.

Elhariri et al. [24] proposed a crack detection model for historical building images based on feature extraction, a fusion of handcrafted features (i.e., HOG and LBP features), and convolutional-neural-network-learned features, and dimensionality reduction using PCA and machine learning classifiers.

In this paper, several concrete bridge crack classification schemes based on histograms of oriented gradients (HOG), uniform local binary patterns (ULBPs), kernel principal component analysis (KPCA), and machine learning classifiers (i.e., SVM, random forests, and decision trees) are studied and compared. The main contributions of this work are the following:

- More than 3000 images from a crack dataset constructed by the authors in [25] are preprocessed using a median filter to remove crack-like noise;
- HOG and ULBP features are extracted from the preprocessed images, and dimensionality reduction by KPCA is applied to the extracted features;
- Classification schemes based on the reduced HOG and ULBP features and machine learning classifiers are investigated and evaluated using different classification metrics to yield the best model for concrete bridge crack detection.

The remainder of the paper is organized as follows: Section 2 briefly explains the HOG and ULBP descriptors and the KPCA dimensionality reduction technique. Section 3 presents the details of the experimental setup followed to implement the classification schemes studied in this paper. Finally, results are presented and discussed in Section 4, and the conclusions of the present work are provided in Section 5.
2. Materials

This section presents a brief explanation of the HOG and ULBP descriptors and the KPCA dimensionality reduction technique used in this paper.

2.1. Histogram of Oriented Gradients

The histogram of oriented gradients (i.e., HOG) [26] is a feature descriptor used to describe edge gradient distribution and orientation.

The vertical and horizontal gradients of an image are calculated by applying convolution filters that are commonly used to find edges in an image. The magnitude and the direction of the gradients are computed following Equations (1) and (2), respectively:

\[ G = \sqrt{g_x^2 + g_y^2} \]  
\[ \theta = \arctan\frac{g_y}{g_x} \]

where: \( g_x \) represents the gradient in the x direction, and \( g_y \) represents the gradient in the y direction.

The image is divided into small cells, and the HOG of each cell is calculated. This histogram is split into nine bins corresponding to angles from 0–180 degrees and shows the frequency distribution of the gradient orientations. The contribution of each pixel within the cell is weighted by its gradient magnitude. The cells are combined into blocks, and the HOG features of the image are generated by combining all of the resulting block features. Figure 1 presents the HOG features of four concrete images with cracks from the dataset constructed by the authors in [25].

![HOG features of concrete crack images from [25].](image)

2.2. Uniform Local Binary Patterns

Local Binary Patterns (LBP) is a texture operator used in various computer-vision applications. Each window in an image is processed to extract an LBP code. The processing includes labeling the center pixel of the window by thresholding its neighboring pixels and generating a binary code following Equation (3) [24]:

\[ LBP_{R,P} = \sum_{i=0}^{P-1} s(g_i - g_c)2^i \]  

where:

\[ s(x) = \begin{cases} 
1 & \text{if } x \geq 0 \\
0 & \text{if } x < 0 
\end{cases} \]
P represents the number of surrounding pixels, R represents the circle radius, and $g_c$ and $g_i$ represent the gray level of the center pixel and the neighboring pixels, respectively.

The generated binary code is converted to a decimal format, and an LBP-based descriptor is created by computing a histogram over the LBP values.

An LBP pattern is called uniform (ULBP) if it has two or fewer transitions when traversed circularly. A unique label is assigned for each uniform pattern, while nonuniform patterns are assigned with the same label.

Figure 2 illustrates ULBP texture features of four concrete crack images.

\[
\text{LBP}_{0,0} = \sum s (g_j - g_0) 2^j = 0
\] (3)

where:
\[s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}\]

2.3. Kernel Principal Component Analysis

Principal component analysis (PCA) is an unsupervised decomposition algorithm used for feature extraction and dimensionality reduction. It performs a linear transformation of features from a higher dimensional space to a lower dimensional space and retains the linear principal components, which maximizes the variance of the data in the low-dimensional space. Generally, the eigenvalues and eigenvectors of the covariance matrix of the original data are calculated, and the eigenvectors represent the new directions in the new feature space.

Kernel principal component analysis (KPCA) is a nonlinear extension of PCA. The data is first mapped to a nonlinear feature space using nonlinear kernels (e.g., radius basis function, sigmoid function, and polynomial function), and the principal components are extracted in the new space.

Figure 3 shows an example of a crack image reconstruction after KPCA application using the radius basis function (RBF) kernel and different numbers of components.
3. Experimental Setup

The diagram in Figure 4 shows the general framework of the experimental setup followed in this paper.

3.1. Dataset

A total of 1304 cracked and 1806 uncracked RGB images at a resolution of 200 × 200 were extracted from the dataset constructed by the authors in [25]. These images cover different representations of cracks and concrete surface in the bridge inspection real world. Figure 5 presents examples from the dataset studied in this paper.

The images were converted to grayscale and preprocessed using a 3 × 3 median filter to reduce crack-like noise.

3.2. Feature Extraction and Dimensionality Reduction

The HOG and ULBP descriptors were used to generate the feature sets from the images of the studied dataset, and dimensionality reduction was performed by applying KPCA. The hyperparameters of the HOGs, ULBPs, and KPCAs were set according to Table 1.

Figure 3. Example of image reconstruction after KPCA application with different numbers of components (original RGB image taken from [25]).

Figure 4. Experimental setup.
Figure 5. Sample images of the studied dataset: Rows (1) and (2) present different patterns of cracks in concrete; rows (3) and (4) present different representations of concrete surfaces.

Table 1. HOG, ULBP, and KPCA hyperparameters.

|        | HOG       | ULBP      | KPCA     |
|--------|-----------|-----------|----------|
| Orientations | 9 (16,16) | (2,2) | 1 | 8 | RBF | 0.00023 | 100 |

A standard scaler was applied to the HOG and ULBP features to normalize them (i.e., the transformed data had a mean equal to 0 and a standard deviation equal to 1) before reducing their dimensions using KPCA.

A reconstruction error based on the root-mean-square error was defined following Equation (4) so as to measure the performance of the dimensionality reduction technique:

\[
\text{Reconstruction Error} = \sqrt{\frac{\sum_{i=1}^{n} (Y - X)^2}{n}}
\]  

where \(X\) represents features before applying KPCA, \(Y\) represents reconstructed features after applying KPCA, and \(n\) represents the number of images.

To further visualize the HOG and ULBP features after the KPCA application, the authors employed the uniform manifold approximation and projection (UMAP) algorithm [27], which is a manifold learning and dimension reduction algorithm effective for cluster visualization.

3.3. Image Classification

Image classification was performed using three machine learning classifiers (i.e., SVM, random forests, and decision trees).

The HOG and ULBP features and the reduced feature sets resulting from dimensionality reduction using KPCA were fed to the three classifiers. Five classification schemes were tested in this work:
• The HOG features were extracted from the filtered grayscale images and were directly fed to the three classifiers;
• KPCA was applied to the extracted HOG features and the new feature set was fed to the three classifiers;
• The ULBPs were extracted from the filtered grayscale images and were directly fed to the three classifiers;
• KPCA was applied to the extracted ULBP features, and the new feature set was fed to the three classifiers; and
• KPCA was applied to the extracted HOG and ULBP features, and the new feature sets were fused and fed to the three classifiers.

The performance of the classification methods was evaluated using a fivefold cross-validation procedure based on the following metrics:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{6}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{7}
\]

\[
\text{F1-Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} \tag{8}
\]

where TP (true positives) refer to the number of cracked images that are correctly classified as cracked; TN (true negatives) refer to the number of noncracked images that are correctly classified as noncracked; FP (false positives): refer to the number of noncracked images that are incorrectly identified as cracked; and FN (false negatives) refer to the number of cracked images that are incorrectly identified as noncracked.

All of the experiments in this paper were conducted in Google Colaboratory with the 12 GB NVIDIA Tesla K80 GPU provided by the platform. The scikit-learn library [28] and the scikit-image Python package [29] were used to implement the classification methods discussed in this paper.

4. Results and Discussions

Table 2 shows the reconstruction error of the HOG and ULBP features using KPCA. It can be seen that the reconstruction errors of HOG and ULBP features using the nonlinear dimensionality reduction technique were significantly low (i.e., an error of $2.82 \times 10^{-8}$ and $1.35 \times 10^{-7}$ for the HOG and ULBP features, respectively). This means that KPCA based on the RBF kernel is capable of reconstructing the HOG and ULBP features and can capture maximum information with a low number of principal components (i.e., 100 components).

| Features | Initial Number of Features | Number of Features after KPCA Application | Reconstruction Error |
|----------|-----------------------------|------------------------------------------|----------------------|
| HOG      | 4356                        | 100                                      | $2.82 \times 10^{-8}$ |
| ULBP     | 40,000                      | 100                                      | $1.35 \times 10^{-7}$ |

Tables 3 and 4 display the accuracy, precision, recall, and F1-score results of the classification schemes based on the HOG and ULBP features, respectively. The SVM model yielded the best classification results as compared to the random forest and decision tree models (e.g., an accuracy of 98.49%, 90.90%, and 72.60% achieved by SVM, random forests, and decision trees, respectively, for the HOG features and an accuracy of 93.44%, 90.10%, and 68.58% achieved by SVM, random forests, and decision trees, respectively, for the ULBP
features). It can also be seen that the HOG features-based classification scheme achieved better results compared to the ULBP texture-based features.

Table 3. Accuracy, Precision, Recall, and F1-scores using HOG features.

| Classification Method          | Number of Features | Accuracy  | Precision | Recall  | F1-Score |
|-------------------------------|--------------------|-----------|-----------|---------|----------|
| HOG + SVM                     | 4356               | 98.49%    | 97.10%    | 99.39%  | 98.22%   |
| HOG + Random Forests          | 4356               | 90.90%    | 88.80%    | 89.65%  | 89.20%   |
| HOG + Decision Trees          | 4356               | 72.60%    | 67.72%    | 66.41%  | 67.03%   |

Table 4. Accuracy, Precision, Recall, and F1-scores using reduced HOG features.

| Classification Method          | Number of Features | Accuracy  | Precision | Recall  | F1-Score |
|-------------------------------|--------------------|-----------|-----------|---------|----------|
| HOG + KPCA + SVM              | 100                | 98.81%    | 98.11%    | 99.10%  | 98.60%   |
| HOG + KPCA + Random Forests   | 100                | 95.40%    | 96.41%    | 92.49%  | 94.10%   |
| HOG + KPCA + Decision Trees   | 100                | 89.20%    | 87.28%    | 86.96%  | 87.10%   |

Figure 6 presents the UMAP visualization maps of the HOG and ULBP features after the KPCA application. It can be noticed that the samples represented by the reduced HOG and ULBP features can be separable, which is convenient for the classification models in differentiating cracked and uncracked concrete images.

Tables 5 and 6 show the classification results based on the reduced HOG and ULBP features after the KPCA application. It can be noticed that the classification metrics were significantly enhanced after the dimensionality reduction of the HOG and ULBP feature sets. This is particularly observable for the random forest and decision tree classifiers (e.g., an accuracy of 95.40% and 90.90% achieved by the random forests classifier, with and without KPCA, respectively, for the HOG features and an accuracy of 93.15% and 90.10% achieved by the random forests classifier, with and without KPCA, respectively, for the ULBP features). The SVM classifier provided the highest accuracies using the reduced HOG and ULBP features (i.e., an accuracy of 98.81% and 93.63% for the HOG and ULBP features, respectively). These experimental results show that, by using only 100 components, the original 4356 HOG features and 40,000 ULBP features are well represented in the nonlinear, new feature subspace, enabling better performance of the three classifiers by addressing and capturing complex data patterns.
Table 5. Accuracy, Precision, Recall, and F1-scores using ULBP features.

| Classification Method    | Number of Features | Accuracy  | Precision  | Recall    | F1-Score  |
|--------------------------|--------------------|-----------|------------|-----------|-----------|
| ULBP + SVM               | 40,000             | 93.44%    | 88.32%     | 97.31%    | 92.58%    |
| ULBP + Random Forests    | 40,000             | 90.10%    | 93.87%     | 81.67%    | 87.31%    |
| ULBP + Decision Trees    | 40,000             | 68.58%    | 62.10%     | 64.65%    | 63.33%    |

Table 6. Accuracy, Precision, Recall, and F1-scores using reduced ULBP features.

| Classification Method          | Number of Features | Accuracy  | Precision  | Recall    | F1-Score  |
|-------------------------------|--------------------|-----------|------------|-----------|-----------|
| ULBP + KPCA + SVM             | 100                | 93.63%    | 89.15%     | 96.62%    | 92.72%    |
| ULBP + KPCA + Random Forests  | 100                | 93.15%    | 89.93%     | 94.25%    | 92.03%    |
| ULBP + KPCA + Decision Trees  | 100                | 90.10%    | 88.56%     | 87.73%    | 88.13%    |

Figure 7 presents the UMAP visualization map of the HOG and ULBP features after KPCA application and feature fusion. It can be seen that the fusion of the reduced HOG and ULBP features resulted in a better separability of the studied samples and would presumably improve the performance of the classification models.

Table 7 summarizes the performance metrics of the three machine learning models based on the fusion of the reduced HOG and ULBP features using KPCA. A gain in performance of the three classifiers was achieved by performing a feature-level fusion after the KPCA application (e.g., an accuracy of 94.24%, 89.20%, and 90.10% achieved by the Decision Tree model with feature fusion, with reduced HOG features and with reduced ULBP features, respectively). It can also be noticed that the SVM classifier yielded the highest classification metrics obtained in this work (e.g., an accuracy of 99.26%) in the classification scheme based on feature fusion after dimensionality reduction of the HOG and ULBP features using KPCA. The fusion of the HOG and ULBP handcrafted features with nonlinear dimensionality reduction provides a strong descriptive capability of complex data patterns. As a result, the proposed classification scheme can accurately separate cracked and uncracked images.
### Table 7. Accuracy, Precision, Recall, and F1-score feature fusion following KPCA application.

| Classification Method                          | Number of Features | Accuracy | Precision | Recall   | F1 Score  |
|-----------------------------------------------|--------------------|----------|-----------|----------|-----------|
| ULBP + HOG + KPCA + SVM                       | 200                | 99.26%   | 99%       | 99.23%   | 99.12%    |
| ULBP + HOG + KPCA + Random Forests           | 200                | 96.21%   | 95.10%    | 95.93%   | 95.50%    |
| ULBP + HOG + KPCA + Decision Trees           | 200                | 94.24%   | 92.93%    | 93.41%   | 93.15%    |

5. Conclusions

In this work, experiments were carried out on the nonlinear (i.e., KPCA) transformation of feature sets generated by the HOG and ULBP feature descriptors. The experiments were validated using a concrete crack dataset covering different representations of cracks and concrete surfaces in the real world of bridge inspection. The images of the studied dataset were preprocessed using a median filter to remove crack-like noise. In addition, this paper evaluated and proposes multiple classification schemes based on HOG and ULBP features, dimensionality reduction using KPCA, feature fusion, and three machine learning classifiers (i.e., SVM, random forests, and decision trees). The performance of the models was evaluated based on standard classification metrics. Experimental results show that applying KPCA improved the performance of the trained models on the HOG and ULBP features as the nonlinear kernel dimensionality reduction technique captures complex data patterns.

Furthermore, the classification scheme based on the SVM classifier and the fusion of the HOG and ULBP reduced features (i.e., using KPCA) provided the best performance with an accuracy of 99.26%. The proposed classification scheme showcases the strong impact of handcrafted feature fusion and nonlinear dimensionality reduction on the performance of classification models in the context of a challenging and complex concrete crack dataset.

**Author Contributions:** Conceptualization, H.Z. and M.R.; methodology, H.Z., M.R. and M.E.A.; software, H.Z.; validation, H.Z., M.R., M.E.A., A.C. and R.S.; formal analysis, H.Z., M.R. and M.E.A.; investigation, H.Z. and M.R.; resources, H.Z.; data curation, H.Z.; writing—original draft preparation, H.Z., M.R., M.E.A., A.C. and R.S.; writing—review and editing, A.C.; visualization, H.Z., M.R., M.E.A., A.C. and R.S.; supervision, M.R.; project administration, M.R.; funding acquisition, A.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** The project received financial support from the NSERC Discovery Grant program (RGPIN-2022-03256).

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Abdel-Qader, I.; Abudayyeh, O.; Kelly, M.E. Analysis of Edge-Detection Techniques for Crack Identification in Bridges. *J. Comput. Civ. Eng.* 2003, 17, 255–263. [CrossRef]
2. Dorafshan, S.; Thomas, R.J.; Maguire, M. Benchmarking Image Processing Algorithms for Unmanned Aerial System-Assisted Crack Detection in Concrete Structures. *Infrastructures* 2019, 4, 19. [CrossRef]
3. Meng, L.; Wang, Z.; Fujikawa, Y.; Oyanagi, S. Detecting cracks on a concrete surface using histogram of oriented gradients. In *Proceedings of the 2015 International Conference on Advanced Mechatronic Systems (ICAMechS)*, Beijing, China, 22–24 August 2015; pp. 103–107. [CrossRef]
4. Meghana, R.K.; Apoorva, S.; Chitkara, Y. Inspection, Identification and Repair Monitoring of Cracked Concrete structure—An application of Image processing. In *Proceedings of the 2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 15–16 October 2018; pp. 1151–1154. [CrossRef]
5. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* 1995, 20, 273–297. [CrossRef]
6. Breiman, L. Random Forests. *Mach. Learn.* 2001, 45, 5–32. [CrossRef]
7. Freund, Y.; Schapire, R.; Abe, N. A short introduction to boosting. *J.-Ipm. Soc. Artif. Intell.* 1999, 14, 1612.
8. Chehri, A.; Saedi, A. IoT and Deep Learning Solutions for an Automated Crack Detection for the Inspection of Concrete Bridge Structures. In *Human Centred Intelligent Systems; KES-HCIS 2021, Smart Innovation, Systems and Technologies*; Zimmermann, A., Howlett, R.J., Jain, L.C., Schmidt, R., Eds.; Springer: Singapore, 2021; Volume 244. [CrossRef]
9. Zhu, J.; Zhang, C.; Qi, H.; Lu, Z. Vision-based defects detection for bridges using transfer learning and convolutional neural networks. *Struct. Infrastruct. Eng.* 2020, 16, 1037–1049. [CrossRef]

10. Yang, Q.; Shi, W.; Chen, J.; Lin, W. Deep convolution neural network-based transfer learning method for civil infrastructure crack detection. *Autom. Constr.* 2020, 116, 103199. [CrossRef]

11. Bukhsh, Z.A.; Jansen, N.; Saeed, A. Damage detection using in-domain and cross-domain transfer learning. *Neural Comput. Appl.* 2021, 33, 16921–16936. [CrossRef]

12. Zoubir, H.; Rguig, M.; el Aroussi, M.; Chehri, A.; Saadane, R.; Jeon, G. Concrete Bridge Defects Identification and Localization Based on Classification Deep Convolutional Neural Networks and Transfer Learning. *Remote Sens.* 2022, 14, 4882. [CrossRef]

13. Choudhary, G.K.; Dey, S. Crack detection in concrete surfaces using image processing, fuzzy logic, and neural networks. In Proceedings of the 2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI), Nanjing, China, 18–20 October 2012; pp. 404–411. [CrossRef]

14. Chen, G.-H.; Ni, J.; Chen, Z.; Huang, H.; Sun, Y.-L.; Ip, W.H.; Yung, K.L. Detection of Highway Pavement Damage Based on a CNN Using Grayscale and HOG Features. *Sensors* 2022, 22, 2455. [CrossRef] [PubMed]

15. Jin, H.; Wan, F.; Ruan, O. Pavement Crack Detection Fused HOG and Watershed Algorithm of Range Image. In *Advances in Internetworking, Data & Web Technologies*; Springer: Cham, Switzerland, 2018; pp. 475–488. [CrossRef]

16. Zalama, E.; Gómez-García-Bermejo, J.; Medina, R.; Llamas, J. Road Crack Detection Using Visual Features Extracted by Gabor Filters. *Comput. Aided Civ. Infrastruct. Eng.* 2014, 29, 342–358. [CrossRef]

17. Chen, F.-C.; Jahanshahi, M.R.; Wu, R.-T.; Joffe, C. A texture-Based Video Processing Methodology Using Bayesian Data Fusion for Autonomous Crack Detection on Metallic Surfaces. *Comput. Aided Civ. Infrastruct. Eng.* 2017, 32, 271–287. [CrossRef]

18. Hu, Y.; Zhao, C.; Wang, H. Automatic Pavement Crack Detection Using Texture and Shape Descriptors. *IETE Tech. Rev.* 2010, 27, 398. [CrossRef]

19. Abdelmawla, A.; Yang, J.J.; Kim, S.S. Unsupervised Learning of Pavement Distresses from Surface Images. In *Advances in Innovative Geotechnical Engineering*; Springer: Cham, Switzerland, 2021; pp. 1–8. [CrossRef]

20. Abdel-Qader, I.; Pashaie-Rad, S.; Abdulayyeh, O.; Yehia, S. PCA-Based algorithm for unsupervised bridge crack detection. *Adv. Eng. Softw.* 2006, 37, 771–778. [CrossRef]

21. Kumar, A.; Kumar, A.; Jha, A.K.; Trivedi, A. Crack Detection of Structures using Deep Learning Framework. In Proceedings of the 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 3–5 December 2020; pp. 526–533. [CrossRef]

22. Endri, E.; Sheta, A.; Turabieh, H. Road Damage Detection Utilizing Convolution Neural Network and Principal Component Analysis. *IJACSA* 2020, 11. [CrossRef]

23. Chen, C.; Seo, H.; Jun, C.H.; Zhao, Y. Pavement crack detection and classification based on fusion feature of LBP and PCA with SVM. *Int. J. Pavement Eng.* 2022, 23, 3274–3283. [CrossRef]

24. Elhariri, E.; El-Bendary, N.; Taie, S.A. Performance Analysis of Using Feature Fusion for Crack Detection in Images of Historical Buildings. In Proceedings of the 11th International Conference on Management of Digital EcoSystems, Limassol, Cyprus, 12–14 November 2019; pp. 308–315. [CrossRef]

25. Zoubir, H.; Rguig, M.; Elaroussi, M. Crack recognition automation in concrete bridges using Deep Convolutional Neural Networks. *MATEC Web Conf.* 2021, 349, 03014. [CrossRef]

26. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893. [CrossRef]

27. McInnes, L.; Healy, J.; Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv* 2020, arXiv:1802.03426. [CrossRef]

28. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 2011, 12, 2825–2830.

29. van der Walt, S.; Schönberger, J.L.; Nunez-Iglesias, J.D.; Boulogne, F.; Warner, J.; Yager, N.; Gouillart, E.; Yu, T. scikit-image: Image processing in Python. *PeerJ* 2014, 2, e453. [CrossRef] [PubMed]