Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 8212790, 22 pages
https://doi.org/10.1155/2023/8212790

Research Article

Enhancing Structural Crack Detection through a Multiscale Multilevel Mask Deep Convolutional Neural Network and Line Similarity Index

Ji-Wan Ham,[1] **Siheon Jeong**,[2] **Min-Gwan Kim**,[2] **Joon-Young Park**,[3] **and Ki-Yong Oh**[2]

[1]*Department of Intelligent Energy and Industry, Chung-Ang University, 84, Heukseok-ro, Dongjak-gu, Seoul 06974, Republic of Korea*
[2]*Department of Mechanical Convergence Engineering, Hanyang University, 222, Wangsimni-ro, Seongdong-gu, Seoul 04763, Republic of Korea*
[3]*KEPCO Research Institute, Korea Electric Power Corporation, 105, Munji-ro, Yuseong-gu, Daejeon 34056, Republic of Korea*

Correspondence should be addressed to Ki-Yong Oh; kiyongoh@hanyang.ac.kr

Received 7 December 2022; Revised 29 July 2023; Accepted 24 August 2023; Published 5 September 2023

Academic Editor: Yu-An Tan

This paper proposes a novel and practical crack-detection method for infrastructure. The proposed method exhibits three key components. First, a multiscale multilevel mask deep convolutional neural network (MSML Mask DCNN) is proposed to accurately estimate crack candidates comprising linear and curvilinear features. Second, the proposed neural network is trained using only public image-sets. The main principle of this approach is that cracks have unique and distinct features, and therefore, public image-sets provide sufficient information to estimate crack candidates for a neural network. Third, a line similarity index (LSI), which is calculated using the Hough transform and coordinate transformation with principal component analysis, is incorporated to eliminate non-crack candidates from crack candidates based on two key characteristics: the variation in crack features with respect to the representative line and the number of crack features that crossed the representative line. Addressing these two crack-related characteristics improves accuracy and robustness by effectively eliminating non-crack features. Field tests performed inside a building and in an underground power tunnel demonstrated the effectiveness of the proposed method. The MSML Mask DCNN outperformed other neural networks, accurately recognizing local crack candidates characterized by linear and curvilinear features even though only public image-sets were used for training. The proposed LSI also effectively eliminated non-crack candidates estimated by the MSML Mask DCNN. The proposed method is practical for real-world applications, where several non-crack objects and noises are typically present.

1. Introduction

Cracks on the surfaces of civil structures are important indicators of defect propagation and structural health. Most defects, including efflorescence, water leakage, exfoliation, and separation, originate from cracks, suggesting that cracks can be considered as representative metrics of structural health [1, 2]. It is important to detect these symptoms in advance because on-time operation and maintenance (O&M) not only decreases the repair cost and time significantly but also prevents the propagation of cracks and degradation of structures. Thus, crack detection is an important initial inspection process in O&M that can effectively maintain the healthy state of civil structures and ensure their safety and reliability.

In recent years, several studies have proposed a variety of crack-detection methods using optical images because optical cameras are inexpensive and can easily record inspection images to evaluate the health state of the surfaces of civil structures. Specifically, cracks in images have distinct features, such as linear or curvilinear shapes, and appearances that are darker than the background. Moreover, cracks are

characterized by the specific appearance of edges and continuity of lines. These features are difficult to be identified by using a conventional signal process and machine learning methods including particle filter (PF) and Gaussian process regression (GPR), which are widely used for predicting system responses [3, 4]. Also, continuity of lines is not easy to be predicted using long short-term memory (LSTM) and broad learning system (BLS) [5]. Therefore, these features can be extracted through image processing methods including image filtering, edge detection, image segmentation, and a hybridization of these methods [6]. However, practical crack detection through traditional image processing is disturbed by noise and low contrast [7]. It is difficult to extract cracks from the background in images with low contrast using image segmentation because the pixel intensities of the crack and background are similar. Furthermore, certain cracks captured in noisy images may be eliminated through noise filtering using traditional image processing techniques. However, most methods merely work on specific images and cannot be commonly applied to all images because the selected hyperparameters for noise filtering depend on the recording conditions. Consequently, specific image processing methods corresponding to the image status are indispensable in the traditional approach [8].

The novelty and key contributions of this study are as follows.

(i) An integrated framework was proposed for effective crack detection in inspection images for real-world applications. This framework comprises a multiscale multilevel mask DCNN (MSML Mask DCNN) and image processing using a line similarity index (LSI).

(ii) The MSML Mask DCNN estimates crack-related pixels from the input image with high accuracy and robustness. Moreover, the MSML Mask DCNN is only trained using public image-sets, implying that no additional effort is needed to record images for training the proposed neural network. The core principle of this approach is that cracks have distinct features, and thus, feature maps from public image-sets provide sufficient information for crack detection.

(iii) The LSI is proposed to exclude non-crack candidates that are classified from the crack estimation process based on two important criteria. One is the deviation of the crack features with respect to the representative line, and the other is the number of crack features that cross the representative line. These two characteristics of cracks ensure high precision and robustness in removing non-crack features.

(iv) The proposed image processing method successfully reduced noise from the acquired images caused by vibration and out-of-focus blurs, which are common phenomena in recorded images.

(v) The proposed framework was validated using public image sets and newly recorded image sets from the inside of a building and from an underground tunnel. A test set of public image sets confirmed the superiority of the proposed MSML Mask DCNN in terms of both accuracy and robustness. Images from field experiments demonstrate that the suggested LSI eliminated over 75% of non-crack pixels from the mask estimated by MSML Mask DCNN, confirming that the proposed method is effective for real-world applications.

The remainder of this paper is organized as follows: Section 2 discusses the related work, Section 3 provides the preliminary information, and Section 4 elaborates on the proposed method. In Section 5, we present the experiments conducted to validate the effectiveness of our approach, while Section 6 offers the results and discussion of the image sets measured from field experiments, analyzing the performance and robustness of our method. Finally, the conclusion and future work are presented in Section 7.

2. Related Work

Considerable efforts have been devoted to developing an effective architecture of DCNN for crack detection. One important factor that can affect estimation accuracy originates from conventional annotation. The training image set for a DCNN is usually annotated with bounding boxes that may include several background pixels in addition to those related to crack characteristics. The background images in bounding boxes deteriorate the extraction of distinct features from crack images, resulting in low accuracy and robustness. This limitation of a square bounding box could be overcome by incorporating a mask DCNN, which addresses an annotated mask based on each pixel of the feature [9]. This method uses images that are annotated in a pixel-to-pixel manner; consequently, this instance segmentation process assigns a label to each pixel of an image. The annotation characteristic of mask DCNNs significantly improves the estimation accuracy of crack detection and has promoted current research on several effective mask DCNNs [10–16]. Deeper and wider neural network architectures that are supervised by annotated images in a pixel-to-pixel manner can ensure high accuracy and robustness in crack detection. Several studies have been conducted on classification models of DCNNs at the image-patch level or region on two stage detectors, which are sequentially region localizing and then classifying. Crack and non-crack classification models are trained at the image-patch level by addressing the crack candidate region [13]. A mask regional convolutional neural network (Mask R-CNN) is also combined with the DCNN using two stage detectors [14]. This architecture of a neural network localizes a crack as an object in an image using a region proposal network at the segmentation of pixel-level. However, validation of the proposed method was limited because image availability is limited. A dual-scale CNN classification neural network, which combines GoogLeNet classifier at a large scale (224×224) and ResNet classifier at a small scale (32×32), was also proposed to detect cracks from different scales of input images [15]. However, dual-scale CNN models require a heavy computational effort because this architecture

repeatedly calls up the operation of each CNN model at each scale of the image. While using a single CNN model for both scales may reduce the computational effort, it adversely affects a performance of crack detection.

3. Preliminary

DCNN with the architecture of a two-stage detector enhances detection accuracy in cases where the target object to be detected is obvious. However, cracks in real-world environments are thin and irregularly shaped, suggesting that cracks that are not fully trained by the training image-set might not be detected by a two-stage detector. Therefore, a DCNN with a one-stage detector exhibits better accuracy in real-world applications. In contrast to one-stage detectors, U2-Net employs an autoencoder architecture that consists of an encoder, decoder, and fusion module, all built with the U-net as a fundamental block, which they refer to as the Re-Sidual U-block (RSU) [17]. This architecture allows U2-Net to efficiently extract feature maps from multiscale and multilevel information at each scale of the encoder and decoder, providing it with a faster processing speed than existing models that rely on backbones for feature extraction. However, a limitation of this approach is that it cannot fully transfer the initial input spatial information to the subsequent scale due to the multiscale and multilevel input manner. A DCNN that was trained with the crack candidate regions and combined with local and global feature regions from the speeded-up robust features (SURF) method and the convolutional neural network (CNN) [13], respectively, has an improved prediction accuracy by utilizing both features as complementary components. However, this method depends on the training image-set and pre-processing step, including the binarization of the crack images in SURF. This step can adversely affect a performance of crack detection because the binarization of the crack image in the SURF method can cause an error in the crack image, including a noisy background. Transforming a classification CNN model to a fully convolutional model was proposed to produce coarse output maps to remove the correlation in the scale of the input image [16]. This method does not require several scales of an input image because the feature maps from different scales of images are trained to build the classifier. However, this method is limited in that it cannot classify patterns or noise that are not included in the training image set. This limitation suggests that training all types of patterns or noise, including cracks, is impossible.

According to the literature survey, two-stage models perform well with respect to the detection area of an image-patch or region but require heavy computational efforts. The network fuses results based on features extracted from each level of the U-net. Moreover, these methods still have several limitations, resulting in false detection due to the environment of the training data. In addition, these neural networks also detect non-crack objects, which have features similar to cracks in real-world applications. Specifically, non-crack objects with crack-like shapes, such as tiles, boundary lines between tiles and tile joints, and wall-to-ceiling boundaries in backgrounds, are estimated as cracks through mask DCNNs. These results considerably limit real-world applications of DCNNs. Moreover, capturing images for training neural networks and annotating cracks in images require great effort. This is another hurdle in the use of DCNN for crack detection. These limitations motivated the present study to establish an improved crack-detection method that can be used in real-world applications.

4. Methodology

A complete flowchart of the proposed method comprising four phases is shown in Figure 1. In phase A, a crack mask was estimated from an optical image using the MSML Mask DCNN. The multiscale multilevel architecture of the proposed neural network ensures a high accuracy and robustness. In phase B, the estimated crack masks were preprocessed using morphological image processing, and a contour filter was used for denoising. These masks were modified to have distinguishable curvilinear lines for calculating the LSI.

In phase C, straight lines corresponding to the preprocessed crack masks were calculated through the Hough transform and then transformed to the principal components according to the principal component axis. In phase D, the LSI was calculated based on the straight lines calculated in phase C and the estimated mask. The probability distribution of the LSI for crack features, which was constructed using a public image-set, was used to eliminate line-like candidates from the estimated mask. The detailed processes performed in each phase are described in the following subsections.

4.1. Phase A: Crack Detection Using the MSML Mask DCNN. In this phase, a crack mask was predicted using the MSML Mask DCNN from a measured optical image ((a) in Figure 1). The MSML Mask DCNN was especially addressed to detect cracks because of its high accuracy and robustness among many mask-based DCNN. Specifically, it constructs a high-scale feature map from low-scale input images and a low-level feature map from high-scale input images, which implies that a multiscale architecture can effectively extract different features at different size of images and fuse them to recognize the object of interest. Furthermore, multilevel feature layers conduct elementwise summation of features extracted from shallow and deep feature maps. This elementwise summation overcomes the gradient vanishing problem because shallow features summed by elements to features in a deeper layer conserve the semantic information of objects [18]. Hence, this architecture would be effective to extract features at different level of complexity.

The architecture of the proposed MSML Mask DCNN is shown in Figure 2. This model is designed to extract cracks of varying lengths from input images recorded in different environments. The theoretical basis of this model lies in the incorporation of multiscale layers in both the encoder and decoder, which enables the extraction of both local and global features. The multiscale layers in the shallow part of the encoder detect fine and sharp features including edges

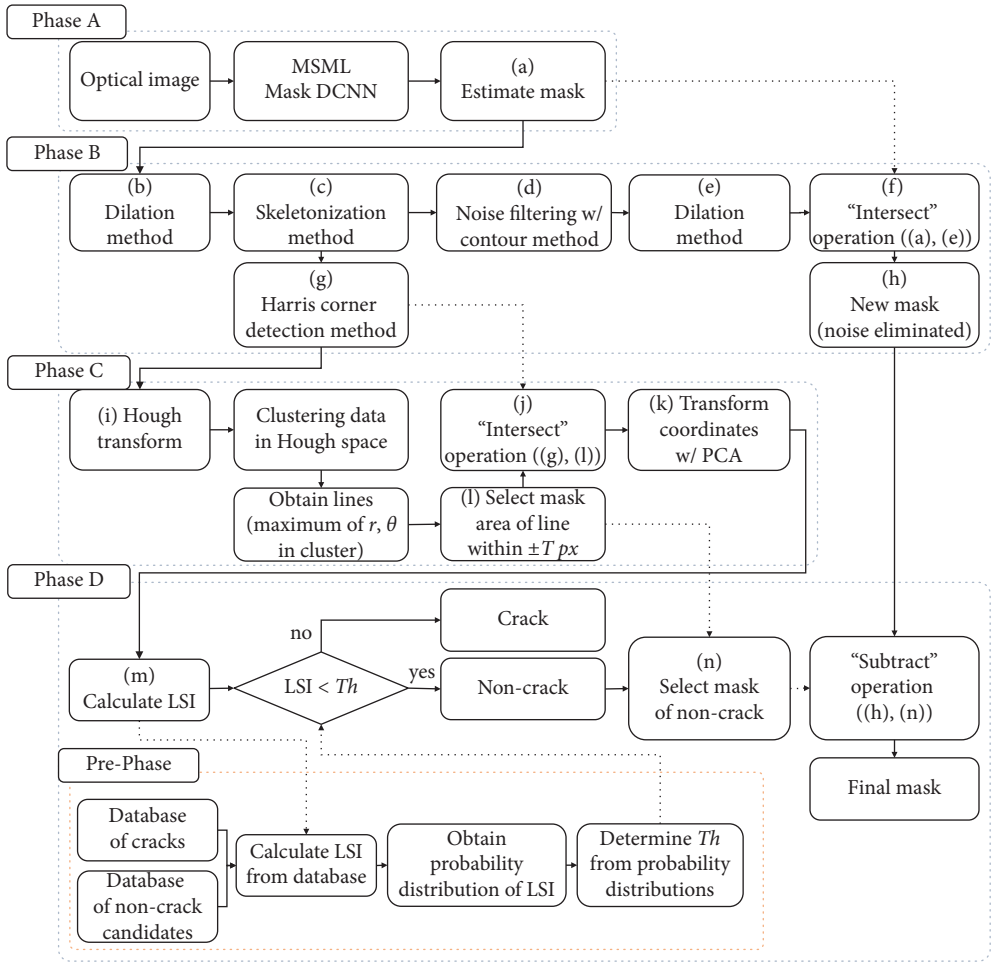


Figure 1: Flowchart of the proposed method.

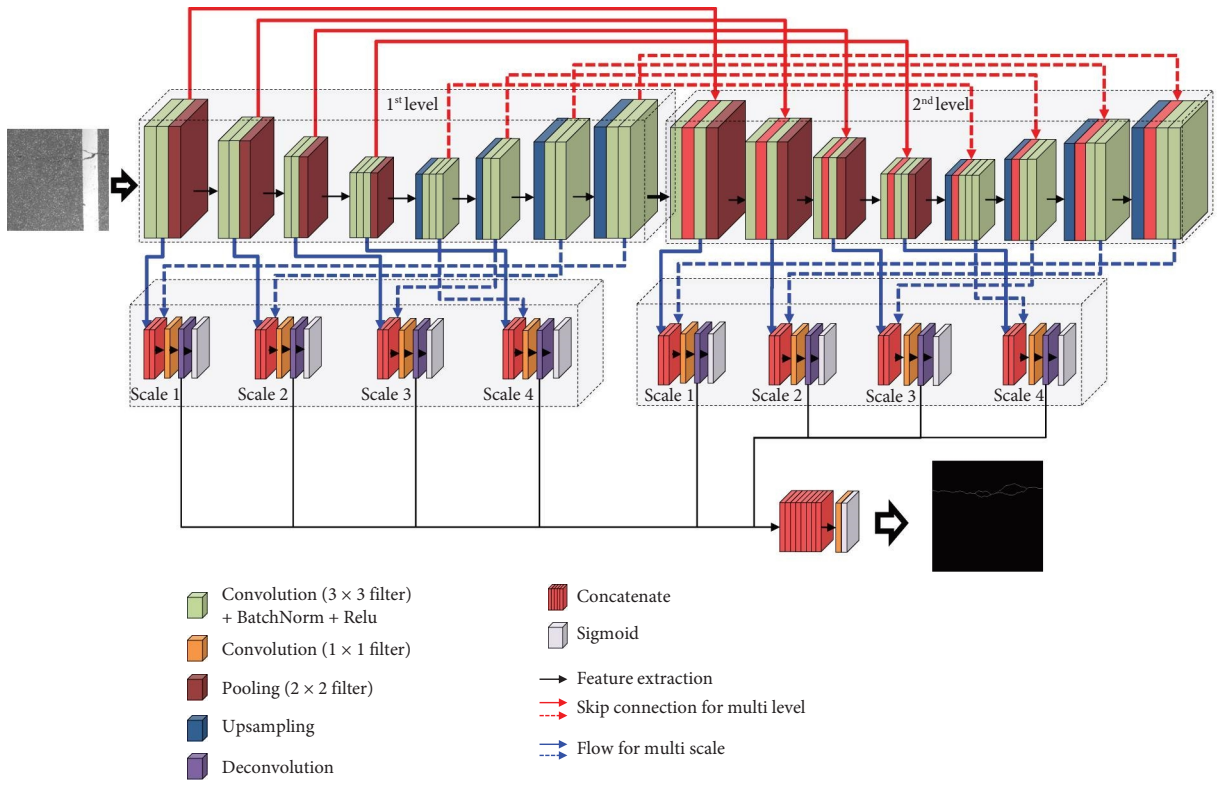


Figure 2: Architecture of the MSML mask DCNN.

and corners, whereas those in the deeper part of the encoder detect smooth yet complex features including surfaces of various objects [19]. The feature maps extracted at each scale are then concatenated to estimate the cracks in an image. This architecture reduces the size of parameters and is computationally efficient [17, 20]. Furthermore, the MSML Mask DCNN includes multiple levels, each consisting of an encoder and decoder pair, with the multilevel encoders capturing multilevel features from shallow to deep. In addition, the MSML Mask DCNN includes multiple levels, each comprising an encoder and decoder pair, with the multilevel encoders capturing features at different levels of abstraction. The encoder which is propagates initial information to lower level helps maintain the spatial information of features, thereby minimizing potential information loss that may occur due to the multilevel autoencoder architecture. The decoder branch adds 1×1 convolution layers after up-sampling and an elementwise sum operation to enhance the learning ability and maintain the smoothness of the features. All outputs in the decoder of the multilevel concatenate the multiscale features of the current level [21, 22]. It is worth noting that the proposed architecture differs from that of U2-Net, even though both address an autoencoder architecture. In essence, the proposed model addresses a cascade form, whereas the U2-Net addresses a nested U-net architecture. Further, the proposed method directly transfers the spatial information of input features into the initial encoder across multiple levels, whereas the U2-Net model sequentially transfers input features through the steps of multiscale and multilevel, which may result in a relatively higher potential for spatial information loss. Hence, the proposed method would be more accurate and robust than other neural networks in the architecture of autoencoder. Quantitative evaluation on performances of the proposed architecture as well as those of other neural networks are described in detail at Section 6.

The features extracted from each layer in the first level execute the convolution operation of the corresponding feature at the same scale and encoder/decoder in the second level extracts the features at each layer of the encoder and decoder. All features extracted from all layers of each scale and encoder/decoder were concatenated to determine the final features of the estimation. Then, the concatenation result executes a convolution operation with a one-by-one filter and a sigmoid function. This process results in a final crack mask. However, the estimated crack mask includes all line-like features including actual cracks and non-crack candidates. Hence, the non-crack mask should be eliminated from the estimated mask for real-world applications.

4.2. Phase B: Preprocessing for LSI Calculation. Phase B aims to not only acquire a new crack mask with denoising ((h) in Figure 1) but also to generate a mask for calculating the LSI ((g) in Figure 1). To achieve these goals, this study utilizes several morphological image-processing methods, including a dilation method, skeletonization [23], and a contour method [24].

First, a dilation method is applied to connect articulation points in the estimated cracks to retain and reinforce the linearity of the cracks ((b) in Figure 1). This method is necessary because an estimated mask for a crack has several disconnected points, even though an actual crack is a long curvilinear line. The dilation operation $\oplus$ is a convolution operation between an input image A and a kernel mask B that performs image dilation, which is formulated as follows:

$$\begin{aligned} A \oplus B &= \left\{ z \mid (\widehat{B})_z \cap A \neq \varnothing \right\}, \\ A \oplus B &= \left\{ z \mid (\widehat{B})_z \cap A \subseteq A \right\}, \end{aligned} \tag{1}$$

where z and $(\widehat{B})_z$ denote the values related to the coordination of kernel mask B and the value of the operation matrix, which is the result of the convolution between A and B transitioned upon z. Second, the subsequent skeletonization method reduces the line thickness of the curvilinear line in the pre-processed mask ((c) in Figure 1). This preprocessing method improves the efficiency of the LSI calculations. Figure 3 shows a flowchart of the skeletonization method comprising opening and erosion operations. The erosion operation $\ominus$ ((c) in Figure 3) is also a convolution operation, as shown below.

$$\begin{aligned} A \ominus B &= \{ z \mid (B)_z \subseteq A \}, \\ A \ominus B &= \left\{ z \mid (B)_z \subseteq \mathrm{A}^{\mathrm{C}} \neq \varnothing \right\}, \end{aligned} \tag{2}$$

where (B) denotes the set of B elements through the operation mask with A and B among z.

Another morphological image processing technique is an opening operation $\circ$, which combines the dilation and erosion operations as follows:

$$A \circ B = (A \ominus B) \oplus B. \tag{3}$$

In the opening operation, the order of methods is important; the erosion operation should be executed first, followed by the dilation operation. Note that the opening operation is an effective method for eliminating small amounts of noise in this study [25]. In the skeletonization method, opening and erosion operations ((b) and (c) in Figure 3) are executed in parallel with an estimated mask. The processed result of the former operation is subtracted from the input mask ((d) in Figure 3) and then united to the processed result of the latter operation ((e) in Figure 3). This process is iterated until the width of the connection node is less than or equal to one pixel, resulting in a thin skeletonized crack line. This mask is input to two subsequent steps, that is, noise filtering ((d) in Figure 1) and the Harris corner detection method ((g) in Figure 1). Noise filtering is used to separate crack and non-crack features, whereas the Harris corner detection method is used to calculate the LSI.

Third, noise filtering is executed by applying a contour method to a pre-processed mask ((d) in Figure 1). A detailed flowchart of the denoising process using the contour method is shown in Figure 4. This method comprises two steps: contour identification and noise filtering. In the first step, a positive pixel, defined as a nonzero pixel, is scanned

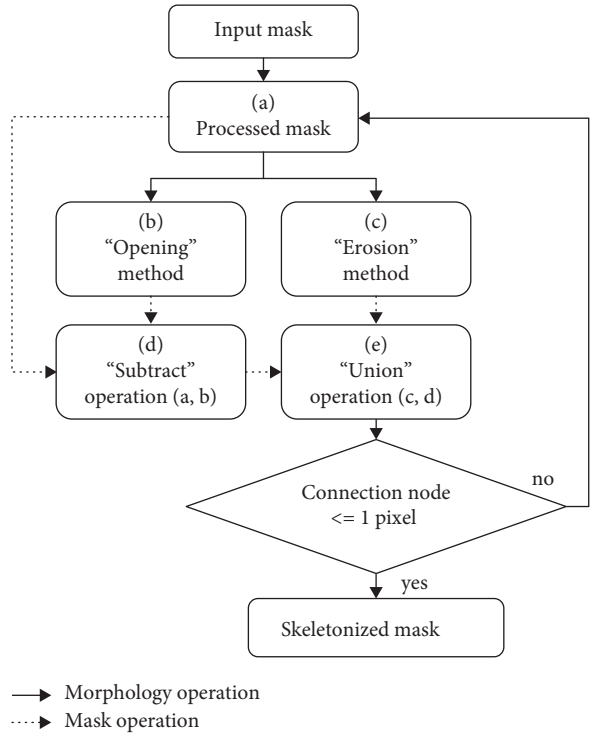


Figure 3: Flowchart of the skeletonization method.

and selected. When a positive pixel is selected, the subsequently connected positive pixels with the same direction of rotation as the initial pixel are identified. The next positive pixel then becomes the center of the points. This process is iterated until the initial positive pixel becomes the center of the points, and the path taken by all tasks is considered the contour of the object. This task is repeated until all pixels of the image have been scanned. In the second step, the area of the contour is calculated; and it is determined if the area is smaller than th_{area}, where th_{area} denotes the predefined threshold of the noise size for elimination. The detected contour is considered as noise if the aforementioned condition is satisfied and eliminated from the input mask. This process is iterated until all contours have been inspected, resulting in a denoised output mask. This mask is input into the second dilation method ((e) in Figure 1), followed by an intersection operation to generate a denoised mask ((f) and (h) in Figure 1). This mask is used for separating cracks from the non-crack features in phase D.

The Harris corner detection method, which is used to enable an effective calculation, [26] is executed in parallel with the previous step to eliminate the intersection point ((g) in Figure 1). The flowchart of the process involved in eliminating the intersection point is shown in Figure 5. The processed mask obtained from the skeletonization method is the input in this method. In this step, a window of size 3×3 was used as a kernel, which was moved along the rows and columns over all pixels. Through this task, (u, v) was calculated as follows:

$$E(u,v) = \Sigma_{(x_i,y_i)\in W}\left[I\left(x_i+u, y_i+v\right) - I\left(x_i,y_i\right)\right]^2, \tag{4}$$

where u and v denote the moving coordinates indicating the row and column inside the window (ranging from -1 to 1), and x_i, y_i denote the row and column coordinates inside the input mask. Furthermore, I denotes the intensity value of the pixel. (u, v) indicates the variation in intensity between the center pixel and one of the other pixels around the center pixel. A value greater than the predefined threshold is regarded as a corner or crossing point. (u, v) can be simplified as follows:

$$E(u,v) = [u\, v] M \begin{bmatrix} u \\ v \end{bmatrix}, \tag{5}$$

where M is defined as follows for efficient calculation:

$$M = \begin{bmatrix} \Sigma\left(\dfrac{\partial I}{\partial x}\right)^2 & \Sigma \dfrac{\partial I}{\partial x}\dfrac{\partial I}{\partial y} \\ \Sigma \dfrac{\partial I}{\partial x}\dfrac{\partial I}{\partial y} & \Sigma\left(\dfrac{\partial I}{\partial y}\right)^2 \end{bmatrix}. \tag{6}$$

Finally, R is calculated from M to classify the size of the singular value in each direction. Then, to determine whether the pixel is a corner, edge, or flat portion, the following calculation is performed:

$$R = \det(M) - k\left(\text{trace}(M)\right)^2, \tag{7}$$

where k denotes the weight of a square trace of M to obtain an appropriate value of R in the range of 0.04 to 0.06. The gap between the R value and zero is calculated. An R value less than zero indicates that the pixel is an edge, whereas an R value close to zero indicates that the pixel represents a flat portion. An R value greater than zero indicates that the pixel represents a corner. Based on this principle, pixels regarded as corner or crossing points are selected. After the pixels have been selected, the region around these pixels is eliminated. If these pixels are not eliminated in this step, the calculated LSI exhibits many errors. The pixels around the crossing point include the information of lines other than the representative line targeted for the LSI. Therefore, this task results in a mask being used for the LSI calculation in the subsequent phases.

4.3. Phase C: Selection of the Representative Crack. In phase C, a straight line representing an estimated mask is first calculated through the Hough transformation ((i) in Figure 1) [27]. Several straight lines are generated for one estimated mask in the Hough space, implying that one representative line will be selected among these candidates. This representative line is selected by utilizing a mean-shift cluster method [28] in the Hough space because this method is faster than other clustering methods and achieves accurate performance with no limitation in the number of detected lines. Note that both processing speed and accuracy are important for real-world applications. The representative line is used to define a candidate region for calculating the LSI, which includes pixels within a predefined distance T with respect to the representative line ((l) in Figure 1). An

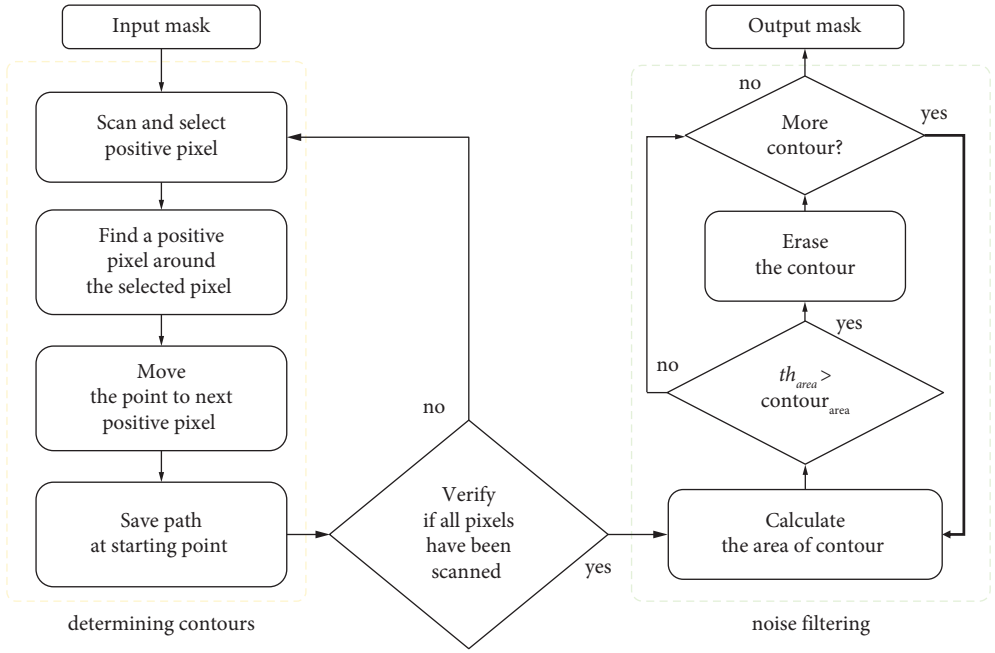


Figure 4: Flowchart of the denoising method.

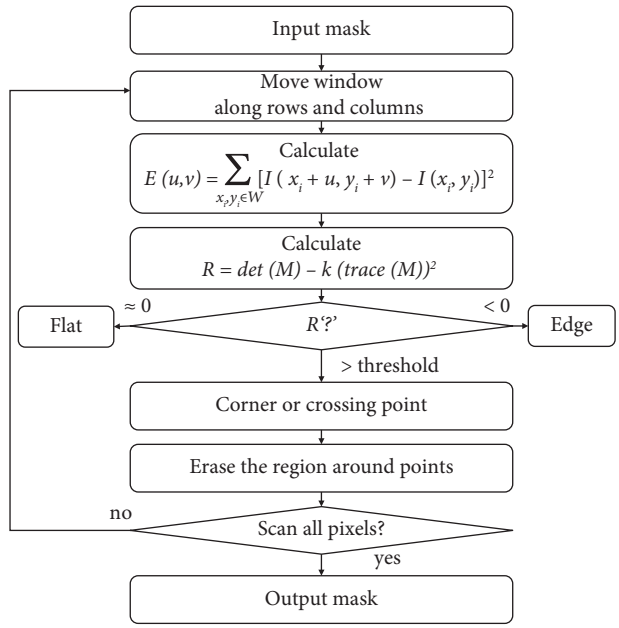


Figure 5: Flowchart for the elimination of the intersection point along with Harris corner detection.

intersection operation is also executed for the mask in this region using the mask created in phase B so that the information available in the original estimated mask is used for calculating the LSI ((j) in Figure 1). When calculating the LSI of the line, the result of the intersection operation ((j) in Figure 1) preserves information regarding the targeted line, and the LSI from this mask is the real targeted information of this phase. Principal component analysis is also performed on the resulting mask in the original coordinate system to determine the two principal axes and to effectively calculate the LSI ((k) in Figure 1). The mask image is then transformed to a hyperplane represented by the first and second axes based on a principal component analysis, such that the x-axis becomes a representative line, that is, the first principal axis through this coordinate transformation, whereas the y-axis represents the second principal axis.

4.4. Phase D: Eliminating Non-Crack Features. In this phase, the LSI is calculated to distinguish actual cracks from non-crack features in the estimated mask ((m) in Figure 1). The LSI is a quantitative metric that represents the variation in

a curvilinear line with respect to the representative line at the principal coordinates, which implies that an actual crack might be characterized by large values of the LSI, whereas line images such as edges and straight line-like objects are characterized by small values of the LSI. This principle is implemented as follows:

$$\mathrm{LSI} = \frac{\sum_{i=x_{\min}}^{x_{\max}} |f(i)|}{x_{\max} - x_{\min}} \times \exp\left(\frac{\alpha n}{x_{\max} - x_{\min}}\right), \tag{8}$$

where x and n denote the first principal axis (x-axis) in the transformed coordinate system and the number of lines in the estimated mask crossing the x-axis, respectively. The subscripts of max and min denote the maximum and minimum values, respectively, of a curvilinear line at the transformed principal coordinates, and f denotes the function formed by the line. The constant α represents the weight of the second term. Note that the LSI combines two crack characteristics. The first is the variation in the crack features with respect to the representative line, which is shown as the first term on the right side of equation (8). The second is the number of crack features that cross the representative line, which is the x-axis in the principal coordinate system. This is the second term on the right-hand side of equation (8). An exponential function with the weight of α is used to ensure an even contribution of each factor while calculating the LSI. Then, the calculated LSI is compared to the probability distribution of the LSI values of cracks, which was created from public crack image-sets. The LSI value of non-crack pixels can be considered an outlier in this probability distribution because the LSI value of non-cracks is smaller than that of cracks. Thus, it is possible to distinguish non-crack features from crack features using the proposed method.

4.5. Evaluation Metrics. Two types of metrics were used in this study for the performance evaluation: metrics for the evaluation of the DCNN and for the proposed method. The former includes the optimal dataset scale (ODS), optimal image scale (OIS), and average precision (AP) [29]. Typical segmentation models use a mean intersection over union (mIoU) metric because it can evaluate clustering or classification performances by comparing the pixels of ground truth and predicted masks [30]. However, the mIoU fails to provide a proper performance evaluation of the prediction result over all images in the image-set because the pixels in the predicted masks are used for the calculation of mIoU, which are determined by a threshold in the classifier. Typically, a threshold of 0.5 is used [31]. In other words, the threshold of the classifier plays a critical role in the evaluation of mIoU. However, this predefined threshold has no regular role as a representative value for the dataset, because the evaluated mIoU up to the threshold value cannot be proportional. This effect stands out in the dataset containing imbalanced objects such as cracks for anomaly detection [32]. Specifically, a crack in an image usually comprises a connection of thin pixels. These unique characteristics are difficult to evaluate using mIoU. Hence, many previous studies related to crack detection have addressed ODS, OIS, and AP instead of mIoU, including DeepCrack, FPHBN, and CrackIT [10, 33–35]. Hence, this study also addressed ODS, OIS, and AP instead of mIoU to evaluate the performance of the proposed method for fair comparison. Note that ODS indicated the best $F1$ score for all thresholds ranging from 0.01 to 0.99 for all image-sets, whereas OIS indicated the best $F1$-score in the same range of thresholds for each image in image-sets. AP indicates the average precision of all image-sets, which is equal to the area of the precision-recall curve. These metrics can be used to compare the performances of the DCNN architecture for crack detection, regardless of the threshold of classification in an imbalanced dataset. Precision is defined as a fraction of the relevant pixels among all the pixels retrieved from the estimated mask. Recall is defined as a fraction of the retrieved pixels among all the relevant pixels obtained from the ground truth. The metrics used to calculate the ODS, OIS, and AP are Precision, Recall, and $F1$-score, which are defined as follows:

$$\begin{aligned} \mathrm{Precision} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \\ \mathrm{Recall} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \\ F1 - \mathrm{score} &= 2\frac{\mathrm{Precision} \bullet \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \end{aligned} \tag{9}$$

where TP, FP, and FN denote true positive, false positive, and false negative rates, respectively, which are calculated by comparing the estimated mask to the ground truth. TP indicates the number of pixels accurately estimated as positive in the estimated mask compared to the ground truth. FP is the number of pixels incorrectly estimated as positive in the estimated mask when compared to the ground truth. FN is the number of pixels incorrectly estimated as negative in the estimated mask when compared to the ground truth.

The metrics used for the evaluation of the proposed method were M_{rem}, M_{elim}, and $F1_M$, which indicate the changes in TP and FN. M_{rem} is defined as the ratio of the remaining TP pixels, that is, $\mathrm{TP}_{\mathrm{rem}}$, to the TP pixels in the estimated mask, which are obtained using the mask DCNN after employing the proposed method, as follows:

$$M_{\mathrm{rem}} = \frac{\mathrm{TP}_{\mathrm{rem}}}{\mathrm{TP}}. \tag{10}$$

M_{elim} is defined as the ratio of the eliminated FP pixels, that is, $\mathrm{FP}_{\mathrm{elim}}$, to the FP pixels in the estimated mask, which are obtained using the mask DCNN after employing the proposed method, as follows:

$$M_{\mathrm{elim}} = \frac{\mathrm{FP}_{\mathrm{elim}}}{\mathrm{FP}} = \frac{\mathrm{FP} - \left(P_{\mathrm{LSI}} - \mathrm{TP}_{\mathrm{rem}}\right)}{\mathrm{FP}}, \tag{11}$$

where P_{LSI} denotes the number of pixels remaining after the deployment of the proposed method.

Both metrics ranged from 0 to 1, where 1 indicated the best performance of the proposed method. Thus, $M_{\mathrm{rem}} = 1$ indicates that all TP pixels remain, and there is no loss in

estimation, and $M_{\text{elim}} = 1$ indicates that all FP pixels are eliminated after deploying the proposed method. However, both metrics demonstrate different aspects related to the performance of the proposed method. Consequently, $F1_M$ was calculated to evaluate the overall accuracy of the proposed method as follows:

$$F1_M = \frac{2M_{\text{rem}} \bullet M_{\text{elim}}}{M_{\text{rem}} + M_{\text{elim}}}. \tag{12}$$

Note that this metric is the harmonic mean of M_{rem} and M_{elim} and thus, $F1_M$ is a representative metric for evaluating the proposed method. However, these metrics should be considered together for quantitative analysis because each metric represents a different aspect of performance. A high value of M_{rem} indicates a greater precision in crack estimation, whereas a high value of M_{elim} indicates that one of the cracks has a greater recall in the evaluation of crack estimation while estimating the performance of the DCNN.

5. Experiments

5.1. Public Image Set. Four sets of crack images obtained from the literature were used for training, validating, and testing the proposed neural network. These four public image-sets, as provided in Ref [10], were acquired and employed for this study. Sample images from each image-set are illustrated in Figure 6, while detailed information regarding all image-sets is presented in Table 1. The CrackTree260 image-set consisted of 260 road pavement images with a resolution of 800×600 pixels, whereas the CRKWH100 and CrackLS315 image-sets consisted of 100 and 315 pavement images with a resolution of 512×512 pixels. Stone331 comprises 331 images of stone surfaces with a resolution of 512×512 pixels. These images were recorded in different environments and presented a variety of cracks of different origins, suggesting that these images are effective for training and testing the accuracy and robustness of the proposed neural network.

As the first step, precise annotation was carried out again to secure accurate ground truths for all image-sets, even though public image-sets already provide ground truths. Although the proposed approach may be executable without further precise annotation, the additional annotation was conducted in light of the dependency of crack detection accuracy on the proposed approach's performance. The provided annotation of crack masks included incorrect and missing annotations for thin cracks and cracks with similar appearances to their environments, making it difficult to quantitatively evaluate neural networks. Our proposed methodology has the ability to determine whether a detected line represents a crack based on the variability of the detected line, which makes additional annotations more useful, especially for detecting minor cracks. This can be stated that this was a valuable task of the proposed framework in phase A ((a) in Figure 1). Note that precise annotation plays a critical role to well train the neural network to cognize crack features including real cracks and crack-like objects with crack-like shapes including tiles, boundary lines between tiles and tile joints, and well-to-ceiling boundaries, whereas the strategy in Figure 1 plays a critical role to remove crack-like objects cognized from the MSML Mask DCNN for field applications. Hence, both tasks are important to inspect cracks in real-world applications. After completing precise annotation, the CrackTree260, CRKWH100, and CrackLS315 datasets were divided into training and validation image-sets. The Stone331 dataset was specifically chosen as the test set to evaluate the impact of training on a particular crack image dataset from one environment on crack detection in different environments. The training and validation image-sets were distributed at a ratio of 9 : 1, resulting in 605 training images and 70 validation images without augmentation (Table 1).

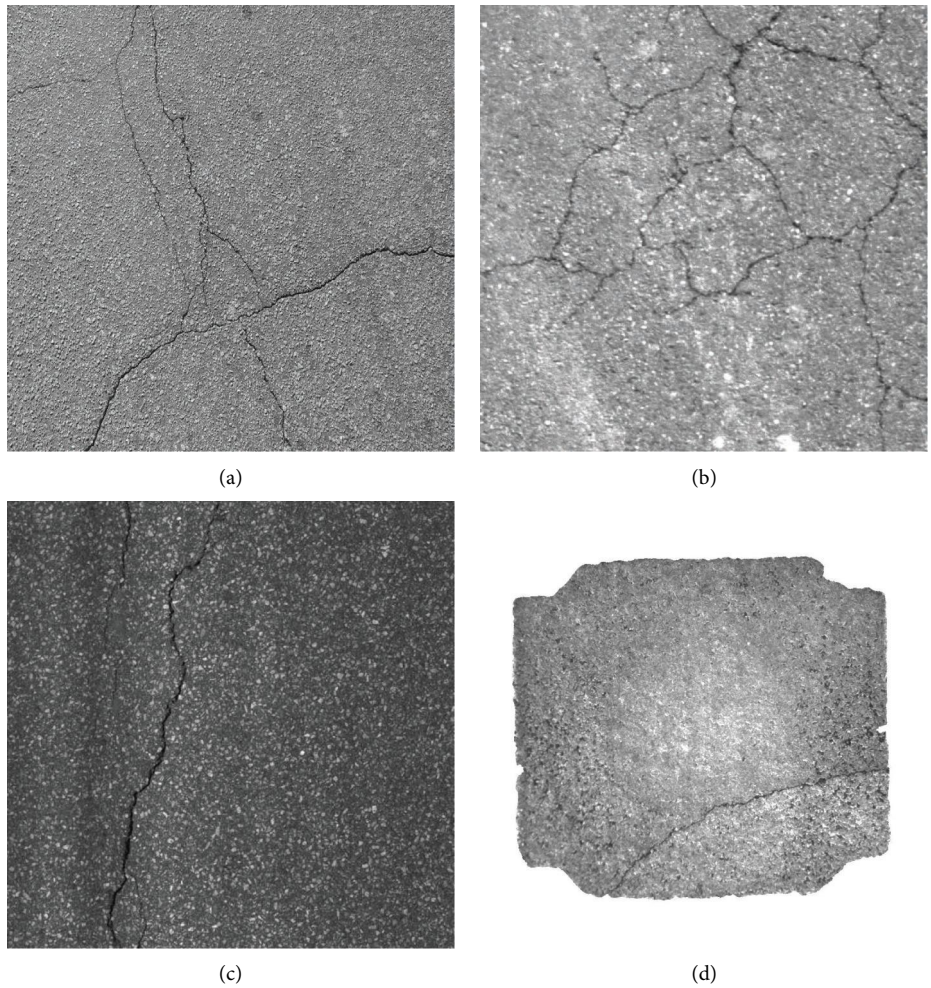
5.2. Field Tests. Two field tests were conducted to demonstrate the effectiveness of the proposed method. One experiment was conducted inside a building, whereas the other was conducted in an underground tunnel equipped with power-transmission facilities. Both structures are made from concrete. The captured images included crack and non-crack features because these mixed features are commonly captured in real-world applications.

The effect of light was first evaluated in field tests because several studies suggested the use of optical and infrared lights for crack measurements. [10, 36, 37] Three types of light intensities (low, medium and high) were used in the measurements under different conditions of light emitting diode (LED) lights. The first case was without an LED; in the second case, the LED, which was fixed on the camera, was facing 45° away from the crack; in the third case, the LED was fixed facing the crack. The images captured under different light intensities were then used as the input images for the MSML Mask DCNN to generate the estimation mask, as shown in Figure 7. Table 2 lists the effectiveness of the different light intensities. Four metrics were used to evaluate the accuracy of the estimated mask (threshold: 0.5). The results clearly indicate that the $F1$-score and AP of the images captured under the high-intensity light showed the best scores of 0.7320 and 0.5430, respectively. This implies that images captured under higher light intensities are effective for accurately evaluating cracks. Hence, all image-sets were recorded under high-intensity light.

In the indoor experiments, crack images were recorded of cracks present on the ceilings and walls of two buildings, numbered 207 and 310, at Chung-Ang University, Seoul. Buildings 207 and 310 were constructed in 1969 and 2016, respectively. The images were recorded using See3-cam_CU135, a 4K USB camera (e-con Systems). The resolution and field of view were set at 1920×1080 and 67°, respectively. This experiment aimed to achieve two goals. First, to obtain the stochastic LSI threshold, which is used for classifying the crack and non-crack features, and second, to test the proposed method. Eighteen images were obtained for determining the LSI threshold. Then, 250 crack and 250 non-crack sub-masks were extracted from these images using a method similar to that used in phase C ((j) in Figure 1). Sub-masks were extracted from the full-resolution image because one image may include two or more target

Table 1: Information of image-sets (original and fused) learning the MSML mask DCNN.

Image-set	(a) CrackTree260	(b) CRKWH100	(c) CrackLS315	(d) Stone331
No. of images	260	100	315	331
Resolution	800×600	512×512	512×512	512×512
Structure		*Pavement*		*Stone surface*
No. of images (training/validation/test)		Training 605 Validation		Test 331
	30	10	30	—



(a) (b) (c) (d)

Figure 6: Public image-sets of a crack. (a) CrackTree260, (b) CRKWH100, (c) CrackLS315, and (d) Stone331.

masks that are line-shaped, and the LSI of each line is calculated. Ten images were considered for testing the model, as exemplified in Figure 8(a). These images include crack and non-crack features, where the non-crack features include tiles on the ceiling or a corner or crossing connected to each wall.

In the underground tunnel, images of structural surfaces such as the ceiling and side wall were captured. The tunnel constructed in 2001, carries Shingwangmyeong-Yeungdeungpo power transmission lines and is managed by the Korea Electric Power Corporation. The O&M protocols for underground power transmission lines recommend that tunnel structures should be inspected bi-annually through a patrol-based inspection. Hence, 2 km of this underground tunnel was inspected. The images shown in Figure 8(b) were acquired using an M5055 pan-tilt-zoom camera (Axis Communications, Sweden) with a 5x optical zoom. The resolution, field of view, and zoom were set to 1920×1080, 71°, and 3x, respectively, to record the images accurately. Twenty images were captured from the underground tunnel. Figure 8(b) shows that these images include crack and non-crack features, confirming that non-crack features in the images should be eliminated to accurately detect cracks. Non-crack features included

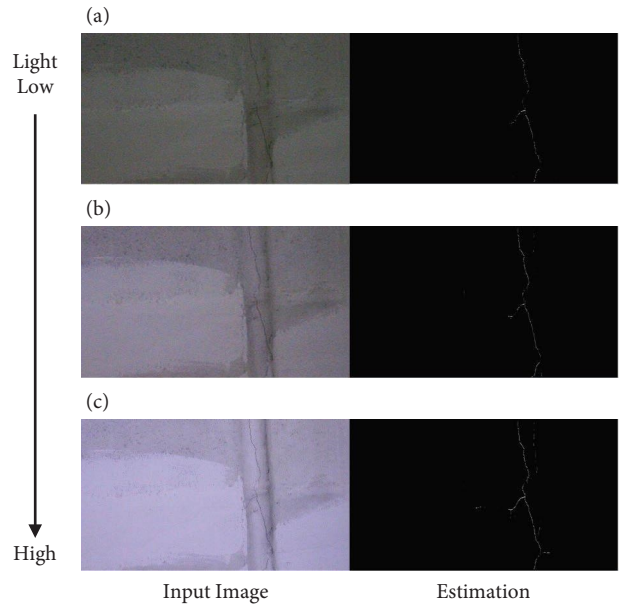


Figure 7: Effect of light intensity on crack measurements (overlapped for visualization by the crack mask with dilation of three). (a) Image taken at the low light and estimated result, (b) image taken at the medium light and estimated result, and (c) image taken at the high light and estimated result.

Table 2: Comparison of the estimated metrics based on the light intensity (@threshold).

Light intensity	Precision@0.5	Recall@0.5	*F*1@0.5	AP
Low	0.9572	0.4611	0.6224	0.4413
Medium	0.9001	0.5664	0.6952	0.5098
High	0.8256	0.6576	0.7320	0.5430

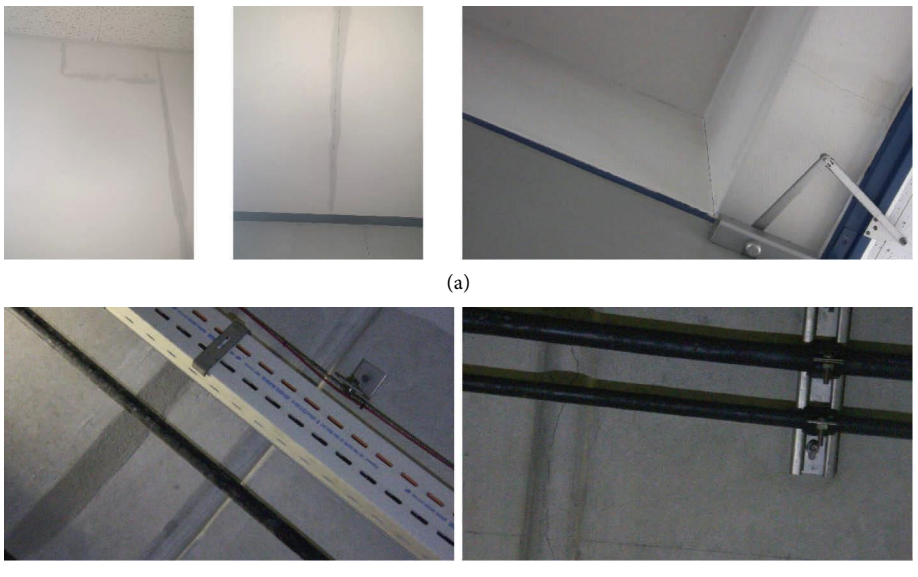


Figure 8: Sample images recorded in field tests. (a) Building and (b) underground tunnel.

transmission lines, hangers, or supports, such as pillars in this experiment. It is worth noting that obtaining multiple images of a crack is difficult because cracks occur randomly and sparsely. This is another reason for using the $F1$-score in the evaluation [38].

5.3. Construction of the Proposed Method. This subsection describes the details for development of the proposed method. This includes the estimation of the hyperparameters used in the MSML Mask DCNN and other variables used in the proposed method.

In phase A, the MSML Mask DCNN and other neural networks were trained, validated, and tested using a graphics processing unit (GPU) server with an Intel Xeon Gold 5218 CPU with 128 GB memory and four NVIDIA RTX 2080 Ti graphics cards. Out of the original image-sets, from CrackTree260, CRKWH100, and CrackLS315, 90% were used for training, and the remaining 10% for validation. All images from Stone331 were used for the test set. The resolution of the input images was reshaped to 512×512 to improve computational efficiency. The MSML Mask DCNN has two levels of the multiscale Mask DCNN (MS Mask DCNN). The MS Mask DCNN levels were constructed by combining three and four different scales of the encoder and decoder networks. Note that fusing the information obtained from three and four feature maps from a deep and wide neural network enhanced the estimation accuracy. The larger the scale of the MS neural network, the more accurate the estimation but the longer the calculation time, suggesting that a trade-off exists in constructing neural networks. Hence, this study uses two different scales of the MSML Mask DCNNs to analyze the advantages and disadvantages of these neural networks in terms of both accuracy and computational effort. A single MS Mask DCNN with the same structure at each level in the MSML Mask DCNN, Mask R-CNN, and Residual UNet (ResUNet) was constructed for quantitative evaluation in terms of accuracy, robustness, and computational load because they show good performances in the literature [39, 40]. The Bayesian optimization (BO) method iteratively evaluates promising hyperparameter configurations within a user-defined budget to achieve the best results [41]. In this study, the BO method was used with a trial number of 100 to optimize the hyperparameters of each model and ensure fairness in the comparison of the different models. The use of BO is important to secure the best results for each model and ensure a fair comparison between the different models. The trial number of the BO method was set to 100 to secure optimal hyperparameters. This study needs to identify the optimal hyperparameters for the early stopping and Adam optimizer, specifically patience, learning rate, β_1 and β_2, and ϵ, when applying the BO method. The BO method should determine the range for each hyperparameter. Hence, this study selects the range of each hyperparameter based on literatures. Specifically, patience, which is the hyperparameter for early stopping, has a value between 10 and 20 [42–44]. However, this study set a relatively broad range of 10 to 50 to account for variations in model size and dataset characteristics. Several studies have indicated that a learning rate of 0.001, β_1 of 0.9, β_2 of 0.999, and an ϵ of 10^{-8} can produce satisfactory results in the Adam optimizer [45, 46]. Based on this recommendation from literatures, β_1 and β_2 were set within the range of 0.9 to 0.999, which are not far from the recommended values. Since ϵ and learning rate have a lesser impact on the results than β_1 and β_2, a wide range was established to explore various combinations of values. Table 3 lists the ranges of hyperparameters used in hyperparameter optimization and the estimated optimal hyperparameters for each DCNN method. Furthermore, as a performance metric for each deep learning model, Frames Per Second (FPS) was calculated for each model's input and output images to compare their performance.

In phase B, the parameter used in the first dilation method was the number of iterations, which was set to three for the steps shown in (b) and (e) in Figure 1. The predefined threshold of the area classifying the noise and cracks, which was determined by the contour method, was set to 50 for the step shown in (d) in Figure 1. The number of iterations used in the second dilation method was set to five. The window block size, aperture size, and threshold of the Harris corner value in the Harris corner detection method were set to 3, 1, and 0.005, respectively.

Phase C comprises the Hough transform to identify the representative lines and coordinate transformation followed by principal component analysis. The Hough transform only has hyperparameters. In the step where the representative line is selected, the width of the pixel around the representative line was set to five for the intersection operation with the result of the Harris corner detector mask.

In phase D, the decision line, which separates crack features from non-crack features, should be chosen to calculate the LSI. The image-sets of the crack and non-crack features comprised 750 and 250 images, respectively. Specifically, 500 images of cracks were randomly selected from the public image-sets listed in Table 1, whereas 250 images of cracks were selected from the images acquired inside the buildings, as mentioned in Section 5.2. Furthermore, 250 images of non-crack features were extracted, as mentioned in Section 5.2. The right-sided three-sigma value was calculated from non-crack features, whereas the left-sided three-sigma value was calculated from crack features. The mean of these boundary values was then considered as the decision line in the probability distribution of the LSI values of the cracks.

6. Results and Discussion

6.1. Superiority of MSML Mask DCNN. The accuracy and robustness of the proposed method were compared to those of other DCNN methods, including Mask R-CNN, ResUNet, and MS Mask DCNN. Public image-sets were used for training, as mentioned in Section 5.1. Table 4 shows the evaluation results in terms of the ODS, OIS, and AP for all validation and test image-sets, and the best scores are shown in bold font for each image-set. Remarkably, the MS Mask DCNN and MSML Mask DCNN show better accuracy than Mask R-CNN and ResUNet, although the calculation speeds

Table 3: Initial range of hyperparameter and optimal value of hyperparameter for each model.

	Hyperparameters				
	Early stopping	Adam optimizer			
	Patience	Learning rate	β_1	β_2	ϵ
	Range of initial settings				
	10~50	10^{-5}~10^{-2}	0.9~0.999	0.9~0.999	10^{-9}~10^{-5}
DCNN methods	Optimal values				
Mask R-CNN	24	5.67e−4	0.924	0.995	1.81e−7
ResUNet	12	1.27e−3	0.922	0.967	5.12e−8
MS mask DCNN w/three scale layers	11	1.46e−3	0.937	0.977	3.57e−7
MS mask DCNN w/four scale layers	18	8.18e−4	0.920	0.990	1.12e−6
MSML mask DCNN w/three scale layers	25	5.50e−5	0.932	0.975	8.94e−8
MSML mask DCNN w/four scale layers	15	8.21e−4	0.915	0.998	1.12e−7

Table 4: Comparison of estimation accuracy for deep convolutional neural networks.

Image-set	DNN method	Metric (%)			FPS
		ODS	OIS	AP	
CrackTree260	Mask R-CNN	87.93	93.85	82.81	24.73
	ResUNet	**97.01**	**97.07**	**94.63**	21.11
	MS mask DCNN w/three scale layers	90.53	90.75	90.95	30.31
	MS mask DCNN w/four scale layers	91.59	91.91	90.16	15.27
	MSML mask DCNN w/three scale layers	91.87	91.75	90.02	21.32
	MSML mask DCNN w/four scale layers	92.66	92.98	91.14	7.05
CRKWH100	Mask R-CNN	89.32	89.43	88.73	25.01
	ResUNet	96.41	96.63	94.24	19.21
	MS mask DCNN w/three scale layers	93.96	94.58	97.59	29.12
	MS mask DCNN w/four scale layers	95.64	96.13	97.66	14.56
	MSML mask DCNN w/three scale layers	96.27	96.31	96.89	20.37
	MSML mask DCNN w/four scale layers	**96.54**	**97.03**	**98.55**	6.91
CrackLS315	Mask R-CNN	84.68	85.98	79.65	25.55
	ResUNet	90.31	87.18	83.39	22.37
	MS mask DCNN w/three scale layers	90.05	88.98	90.39	30.33
	MS mask DCNN w/four scale layers	90.03	88.33	89.75	14.78
	MSML mask DCNN w/three scale layers	90.72	87.73	89.01	21.10
	MSML mask DCNN w/four scale layers	**91.60**	**89.17**	**92.88**	7.02
Stone331	Mask R-CNN	58.73	59.37	47.35	24.21
	ResUNet	83.21	80.11	72.65	22.13
	MS mask DCNN w/three scale layers	81.32	79.54	78.03	30.98
	MS mask DCNN w/four scale layers	82.03	81.38	78.82	14.97
	MSML mask DCNN w/three scale layers	83.54	83.89	80.35	21.34
	MSML mask DCNN w/four scale layers	**84.71**	**84.40**	**83.26**	7.05

(Best scores shown in bold font).

of Mask R-CNN and ResUNet are generally faster than those of the MS Mask DCNN and MSML Mask DCNN. These results are reasonable because the multilevel and multiscale architecture of a neural network increases both the prediction accuracy and computational loads. Specifically, the Mask R-CNN model requires both segmented mask information and bounding box information of the object. However, cracks scarcely fill the bounding boxes owing to their diverse shapes, such as irregularly stretched lines or meshes, and because it is difficult to separate each crack as an object. This DCNN also performs poorly in instance segmentation of cracks, resulting in a relatively low accuracy compared to other neural networks. The ResUNet has a similar structure to the MS Mask DCNN, which is based on the auto-encoder, and the results of the ResUNet were approximately equivalent but marginally worse than those of the MS Mask DCNN and MSML Mask DCNN. Specifically, the mAP of the ResUNet was 90.75% for the validation image-sets of CrackTree260, CRKWH100, and CrackLS315, whereas that of the proposed neural network was 94.19% for three validation image-sets. Moreover, the AP of the ResUNet and MSML Mask DCNN with four layers were 72.65 and 83.26% for the test image-set of Stone331, confirming that the proposed neural network outperforms the

Mask R-CNN and ResUNet. Note that the ResUNet showed better accuracy in the three metrics only for the validation image-set of CrackTree260. This can be explained by the fact that the ResUNet would be overfitted for the CrackTree260 image-set because this image-set was partitioned into training and validation sets. Note also that the phenomenon a neural network can be overfitted to the training image-set is in common and therefore this evaluation of neural network capability focuses on results of the test images-sets in general [47]. The results for the Stone331 image-set clearly indicated that generality and accuracy of MSML Mask DCNN is better than those of the ResUNet because the Stone331 image-set was measured at different environment from the training image-sets. It can be inferred that the ResUNet estimates more false positive pixels than the MS Mask DCNN. This difference originates from the architecture of the neural network that extracts feature maps at each scale layer in the auto-encoder. Specifically, the ResUNet concatenates feature maps corresponding to each scale in the decoder network for augmentation of features and then estimates the mask at the last layer of the network. Consequently, the feature maps of the encoder network are combined with those of the decoder network, and then the feature maps are faded out. In contrast, the MS Mask DCNN utilizes the feature maps together to calculate a loss and to estimate a mask directly. This architecture conserves the semantic information of cracks, which are extracted from different scales, resulting in better performance for detecting cracks. Table 4 also compares the MS Mask DCNN and the MSML Mask DCNN. Interestingly, the MS Mask DCNN and MSML Mask DCNN with two different scale layers show similar accuracy within a maximum of 3% for the three-validation image-sets, even though calculation speed significantly depends on each neural network. This observation can be explained by two hypotheses: First, the weights of neural networks would be overfitted because all images were measured in the same environments, even though validation image-sets were not used for training. Second, the multilevel architecture of the MSML Mask DCNN might not be effective for crack detection because crack features would be simpler than expected.

The first hypothesis is more reasonable than the second hypothesis in this study because the three metrics for the test image-set of Stone331 show different results. The MSML Mask DCNN outperformed the MS Mask DCNN for the same scales of three and four layers. Specifically, ODS, OIS, AP of the MSML Mask DCNN were higher than those of the MS Mask DCNN by 2.22, 4.35, and 2.32% for three scale layers, whereas those of the MSML Mask DCNN were higher than those of the MS Mask DCNN by 2.63, 3.02, 4.44% for four scale layers. These results suggest that the MSML Mask DCNN is more accurate and robust for crack estimation using images measured from different environments. This observation can be explained by the fact that the MSML Mask DCNN not only maintains the advantage of the MS Mask DCNN, but also improves the performance by using feature maps extracted from the second level of the network, which is concatenated to feature maps from the first level of the neural network. The feature maps extracted from each scale in the second level of the network also contributed to the estimation of cracks. Therefore, additional feature maps at each scale in the second level of the neural network, which are based on the feature maps extracted from the first level of the neural network, improve the performance in detecting objects precisely. In conclusion, the MSML Mask DCNN outperforms other neural networks in terms of both accuracy and robustness, even though a complex architecture increases the computational load in estimating cracks. It should be noted that testing the proposed neural network with images from different environments is important to secure estimation accuracy and robustness. However, the FPS of the MSML Mask DCNN was slower than that of the MS Mask DCNN. This trade-off is an important consideration in real-world applications.

6.2. Applications of Each Phase. The results for each phase executed using the proposed method are described in this subsection. One image, which included several non-crack features, was used to demonstrate the effectiveness of the proposed method, and a detailed transformation of this image is shown in Figure 9.

In Phase A, the candidate mask is estimated using the MSML Mask DCNN. Figure 9 (a) shows that an image estimated using the MSML Mask DCNN contained several line and pattern features. Line features originated from the connections between different tiles, and pattern features are obtained from the patterns on the tiles. These features were estimated as crack features because their characteristics are similar to those of cracks. This was a common in real-world applications, which degrades the accuracy of estimating cracks through neural networks.

In Phase B, the dilation method is executed on the estimated mask candidates (Figure 9 (b)). This method connects the disconnected cracks in the mask. The skeletonization method is then applied to the dilated mask (Figure 9 (c)). This process thins the dilated mask, resulting in a realistic image, from which it is easier to select the representative line for the LSI calculation. The size of the patterns, which are considered noise in this study, becomes smaller than the predefined threshold; consequently, the contour method eliminates these patterns, that is, noise (Figure 9 (d)). The mask does not include noise after this step, implying that it can be compared to the original estimation. The dilation method is executed again because certain pixels are disconnected after implementing the prior steps (Figure 9 (e)), and then an intersection operation is performed between the original mask and the dilated mask, resulting in a new mask that only includes line or curvilinear features without patterns or noises (Figure 9 (f)). Simultaneously, the Harris corner detector is applied to the mask obtained from the skeletonization method. Pixels around the

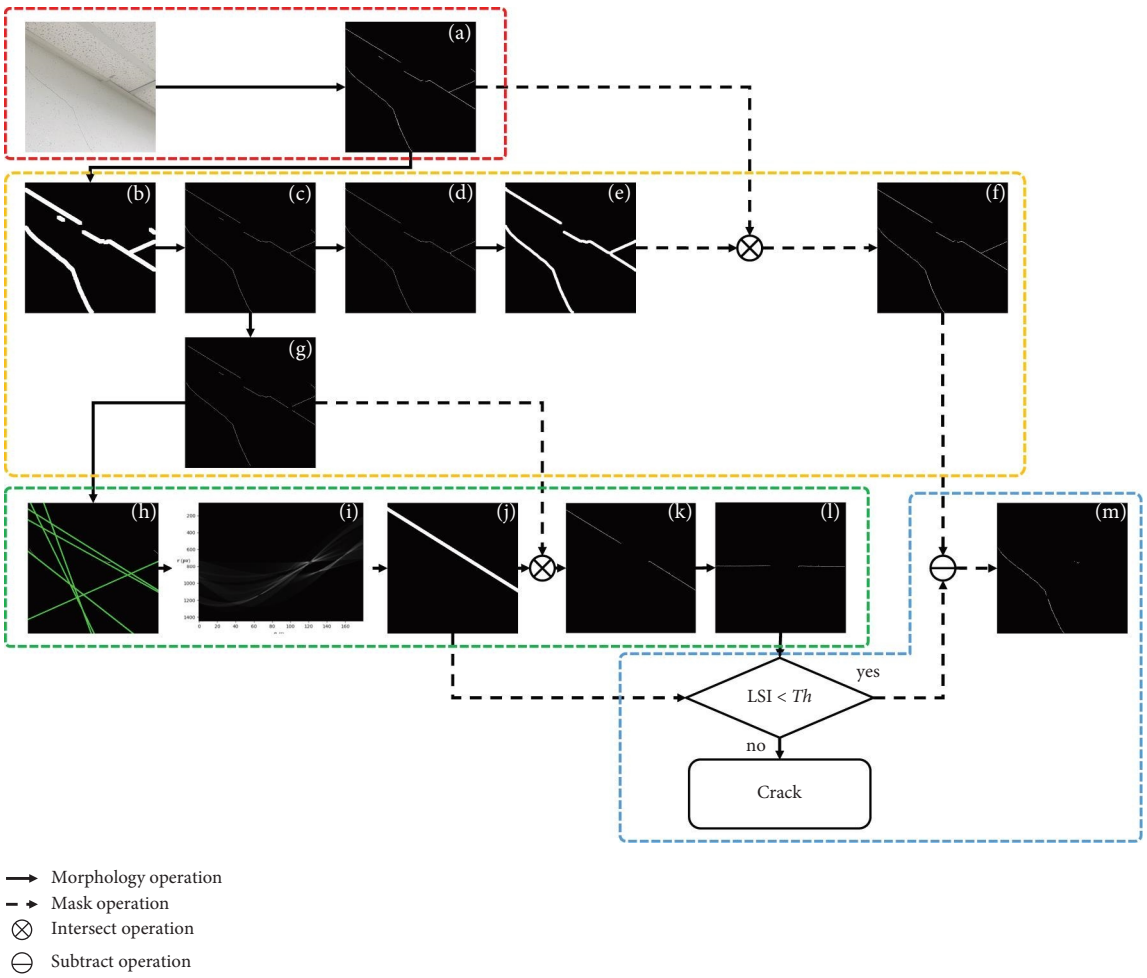


Figure 9: Results of each phase through the proposed method.

corner or crossing points detected by the Harris corner detector are eliminated to minimize the error of the LSI (Figure 9 (g)).

In Phase C, the Hough transform obtains representative lines for the LSI from the mask after the application of the Harris corner detector (Figure 9 (h)). This method transforms all the information in an image into Hough spaces comprising r and θ coordinates (Figure 9 (i)), where r and θ denotes the distance between the origin coordinates in a mask and a line and the angle between the x-axis in a mask and a line, respectively. A line that is thicker than one pixel is selected as a representative line because the LSI is calculated from pixels around the representative line, which is selected based on the Hough transform ((j) in Figure 9). An intersection operation is performed between this mask and the mask from phase B ((g) in Figure 9) to select a targeted mask for the LSI calculation ((k) in Figure 9). Then, coordinate transformation was performed, followed by principal component analysis to analyze the line and curvilinear features in the first principal axis ((l) in Figure 9). These steps, from phases A to C, prepare a mask for the LSI calculation.

In phase D, non-crack features are eliminated based on the probability distributions of the crack and non-crack image-sets. Figure 10 shows this probability distribution with the decision line, which was calculated from the mean of the three-sigma boundary values (99.7%) for crack and non-crack features. Specifically, the three-sigma value of the cracks on the left side is 0.7321, whereas that of the non-cracks on the right side is 0.6719, resulting in a decision line of 0.7 as the mean of these two values. This clear separation of the two features originated from two key characteristics: the variation of crack features with respect to the representative line and the number of crack features that crossed the representative line, suggesting that the decision lines can classify crack and non-crack features. Hence, a subtraction operation in this phase effectively eliminates non-crack candidates in the mask. It is worth noting that the density of the crack features was different from that of the non-crack features. The former is in the range of 0.0 to 0.1, whereas the latter is in the range of 0 to 6, confirming again that the proposed LSI is an effective metric for distinguishing crack features from non-crack features. When the LSI of the representative line was smaller than that of the decision line,

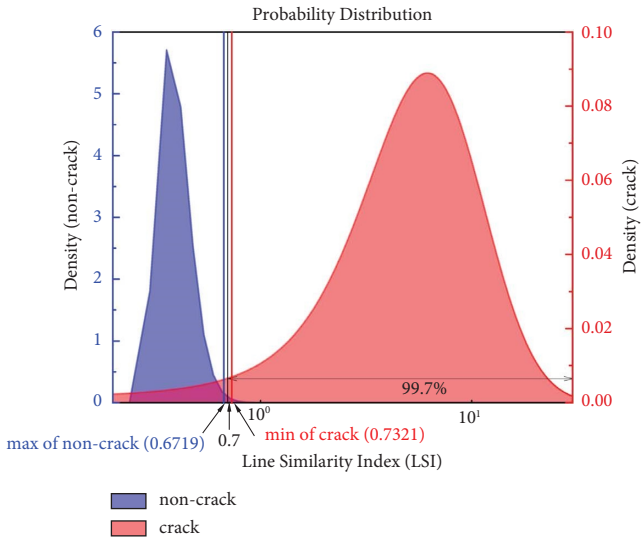


Figure 10: Probability distribution of the LSI.

the representative line mask ((j) in Figure 9) was subtracted from the new mask ((f) in Figure 9). Finally, the subtracted mask becomes the final mask ((m) in Figure 9), where the non-crack components have been eliminated. This process was repeated until the number of representative lines had been estimated.

6.3. Cracks in Building Interiors. Figure 11 shows images after deploying the MSML Mask DCNN and the proposed method. Table 5 lists the three values of metrics evaluating the proposed method with the indoor test image-set. Estimating cracks with the MSML Mask DCNN includes both non-crack pixels, which are components of the ceiling tiles, and crack pixels on the concrete wall (Figure 11 (A-3)), whereas the proposed method eliminates non-crack pixels (Figure 11 (A-4)). Non-crack features were mostly eliminated, confirmed by the values of M_{rem}, M_{elim} and $F1_M$, which are 0.98, 0.94, and 0.96, respectively (image No. 8 as in Figure 11(a)). Specifically, 2% of the estimated pixels among the TP pixels were eliminated, whereas 94% of FP pixels, which are the non-crack components of tiles and notches on ceilings, were eliminated. This analysis demonstrates that the proposed method effectively eliminated non-crack features, whereas remained crack features. The non-crack components eliminated depend on the predefined threshold of the area classifying noise and crack, which is decided by the contour method, suggesting that an appropriate threshold would be important for real-world applications. Noise closely located to the representation line also degrade accuracy, implying that objects around cracks play an important role in detection accuracy. These factors result in low accuracy for some recorded images including image No. 3 and 7. Considering image No. 3, the values of M_{rem}, M_{elim} and $F1_M$ were 0.36, 0.98, and 0.53, respectively, suggesting that TP pixels were mostly not eliminated, whereas FP pixels were almost eliminated. Specifically, cracks located on the inner side of a corner were barely detected because one long crack was detected as several short cracks, and thereby TP pixels were considered noise in the proposed method; please see Figures S1(a) and S1(b), which are enlarged figures of Figure 11 (B-2) and (B-3). An edge of a corner has a line-like characteristic and therefore the corner negatively affected crack detection because the cracks were located around the corner. By contrast, image No. 10 shows low $F1_M$ because FP pixels were mostly not eliminated, whereas TP pixels were almost eliminated. Exact values for three metrics M_{rem}, M_{elim} and $F1_M$ were 0.83, 0.63, and 0.71, respectively.

This result is also highly correlated to the objects around the cracks and the geometrical environment. Specifically, image No. 10 includes several non-crack pixels that are components of the door and ceiling tiles (Figure 11(c)). Figure 11 (C-4) shows that the results of the proposed method retained several non-crack features because of the adjacent lines to the representation line, which have lower pixels than the threshold of the LSI. The proposed method calculates the LSI from an area of crack candidates with respect to the representation line, implying that the LSI is proportional to the size of the amplitude and variation of the crack candidate. If a crack candidate involves a crossing line because of other objects (as in Figure 11 (C-4)), this crossing line results in an error in the LSI calculation because it amplifies the LSI of the crack candidates. The Harris corner detection was used in the proposed method to eliminate the area of the crossing point. However, the Harris corner detection method also defines an appropriate threshold for hyperparameters to detect the corner of crossing points and the results depend on the images. In this case, the crossing points are not fully detected, and this causes an error for a high LSI to be retained as a crack. [48].

Regardless of these limitations, the proposed method shows a good capability for crack detection in real-world applications. To validate the superiority of the proposed method, the accuracy of the proposed method is compared to other state-of-the-art methods in literature [13–16] (Table 6). The architecture of the other four methods was replicated from Ref. [13–16] and then trained with the same public image-sets (Table 1). Finally, the hyperparameters for each method were optimized using the BO method. The results showed that the proposed method outperformed other state-of-the crack detection methods, suggesting that the proposed deep and wide neural network effectively detects crack features, even though the MSML Mask DCNN was only trained using public datasets. Moreover, the AP of the entire framework fusing the MSML Mask DCNN and LSI shows at least two times higher accuracy than the others, confirming that false positive pixels are successfully eliminated by the proposed method. However, the proposed method is slower than other methods because of the deep architecture of the MSML Mask DCNN and LSI processing, suggesting that a high performances GPU would be indispensable for implementing the proposed method in real-world applications. This comparison clearly suggests that the proposed method is effective in real-world applications.

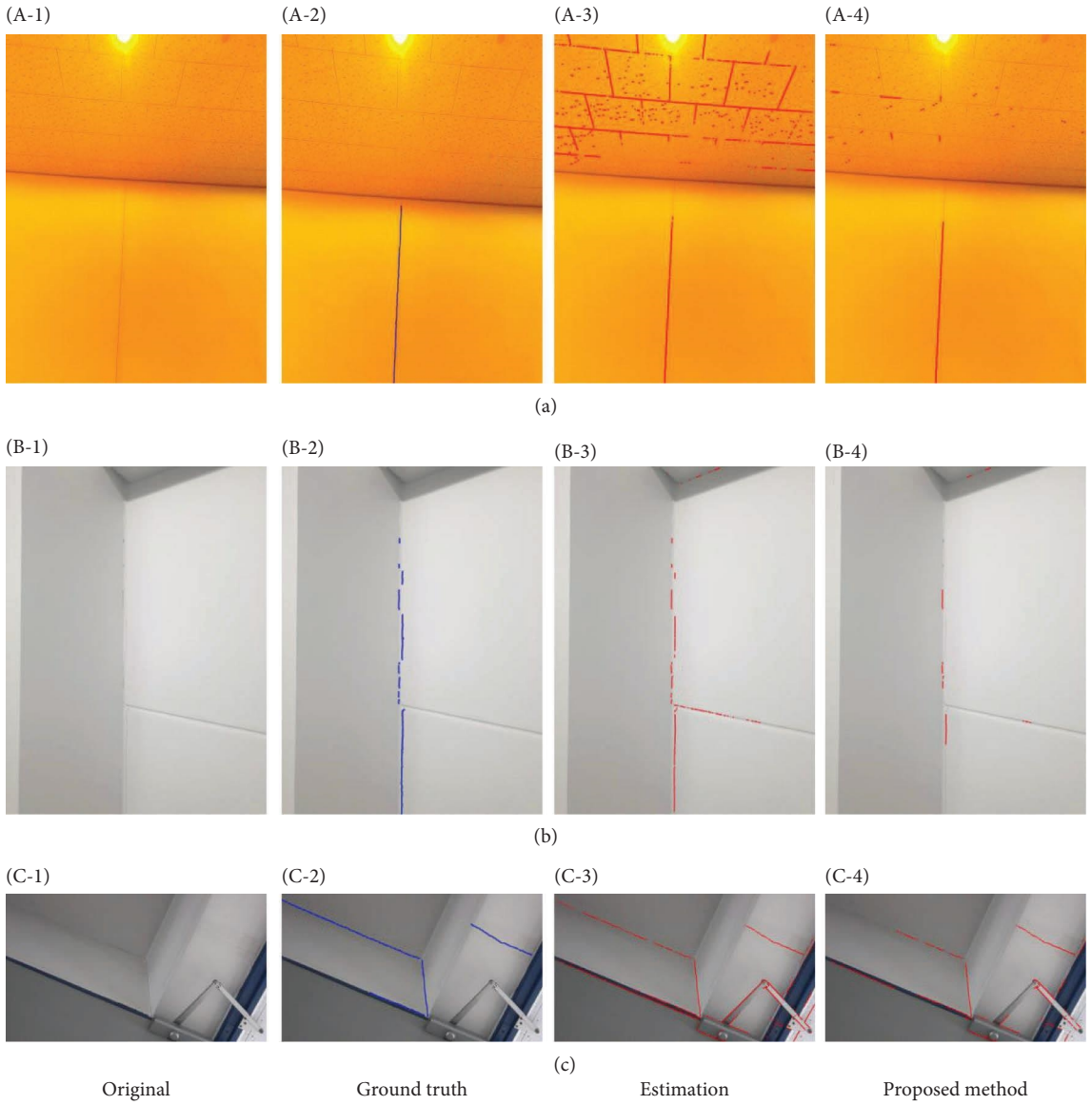


Figure 11: Estimated masks of sample images captured inside a building (overlapped for visualization by the crack mask with dilation of three). (a) to (c) The notation 1 to 4 corresponds to the original image, original image overlapped by ground truth, original image overlapped by the DCNN estimation result, and original image overlapped by the result from the proposed method.

Table 5: Result of the indoor test set (total 10 images) using the proposed method.

Image no	1	2	3	4	5	6	7	8	9	10	Mean
M_{rem}	0.73	0.52	0.36	**0.99**	0.78	0.54	0.37	0.98	0.98	0.83	0.71
M_{elim}	0.98	0.89	0.98	0.92	0.92	0.73	**0.99**	0.94	0.88	0.63	0.89
$F1_M$	0.83	0.65	0.53	**0.96**	0.84	0.62	0.54	**0.96**	0.93	0.71	0.76

(Best scores are shown in bold font).

6.4. Cracks in the Underground Tunnel. Figure 12 shows sample images of applying the proposed method on the underground tunnel image-set and Table 7 lists the quantitative results of the proposed method for the three metrics. Image No 12 shows one of the best results (Figure 12(a)); the metrics M_{rem}, M_{elim} and $F1_M$ are 0.97, 0.94, and 0.95, respectively, suggesting that the proposed method effectively eliminates non-crack features in crack candidates. Similarly, image No. 1 also obtained a high score for the metrics (M_{rem}: 0.99, M_{elim}: 0.95 and $F1_M$: 0.97), confirming again that the MSML Mask DCNN accurately estimates all crack features, and the proposed LSI effectively eliminates non-crack candidates. By contrast, values of M_{rem}, M_{elim} and $F1_M$ were 0.93, 0.19, and 0.32 for image No. 6 (Figure 12(c)), implying that only a few FP pixels were eliminated by the proposed method. Three categories of non-crack features

Table 6: Comparison of estimation accuracy for crack-detection methods.

Image-set	DNN methods	Metric (%)			FPS
		ODS	OIS	AP	
Indoor test set (total 10 images)	MSML mask DCNN w/four scale layers	20.12	21.53	5.39	6.98
	MSML mask DCNN w/four scale layers w/LSI	**29.86**	**29.99**	**12.35**	5.10
	Surf & CNN-based classification model [13]	8.55	6.43	1.98	21.58
	Mask R-CNN [14]	9.58	8.59	2.12	23.63
	Dual-scale CNN-based classification [15] (GoogLeNet & ResNet)	15.73	11.28	4.32	10.23
	CrackPix [16]	13.30	10.19	2.40	15.33

(Best scores shown in bold font).

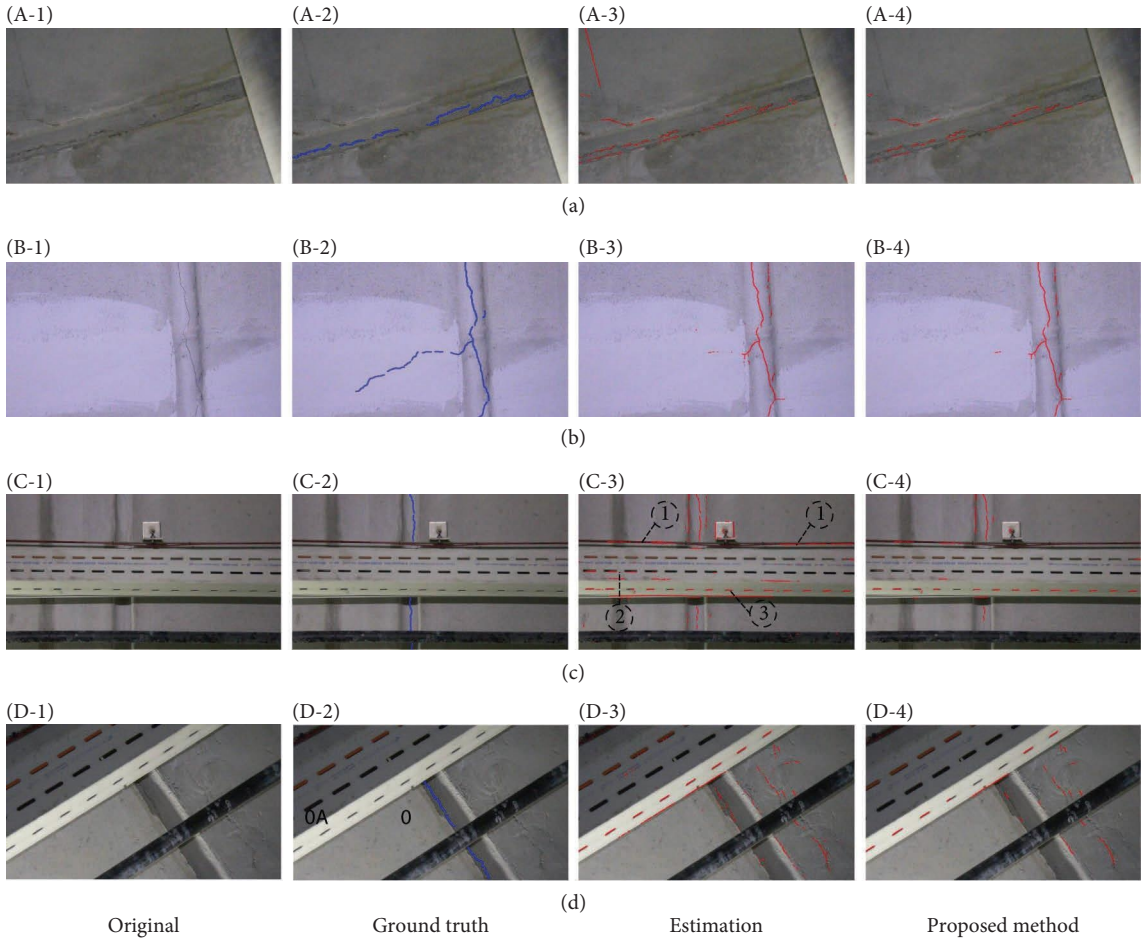


Figure 12: Estimated masks of sample images captured inside a building (overlapped for visualization by the crack mask with a dilation of three). (a) to (d) correspond to images no. 12, no. 1, no. 6, and no. 11. The notation 1 to 4 corresponds to the original image, original image overlapped by ground truth, original image overlapped by the DCNN estimation result, and original image overlapped by the result from the proposed method overlapped on the image.

Table 7: Result of the underground tunnel test set (total 20 images) using the proposed method.

Image no	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Mean
M_{rem}	0.99	**1.00**	0.98	0.97	0.98	0.93	0.99	0.63	0.71	0.98	0.46	0.97	0.97	0.95	0.96	**1.00**	0.98	0.70	0.98	1.0	0.91
M_{elim}	0.95	0.91	0.61	0.95	0.26	0.19	0.51	**0.99**	0.98	0.92	0.72	0.94	0.90	0.73	0.62	0.93	0.26	0.53	0.92	0.45	0.71
$F1_M$	**0.97**	0.95	0.76	0.96	0.41	0.32	0.67	0.77	0.82	0.95	0.56	0.95	0.94	0.82	0.75	0.96	0.41	0.61	0.95	0.62	0.76

(Best scores are shown in bold font).

Table 8: Comparison of estimation accuracy for crack-detection methods.

Image-set	DNN methods	Metric (%)			FPS
		ODS	OIS	AP	
Underground tunnel test set (total 20 images)	MSML mask DCNN w/four scale layers	34.22	35.41	11.88	7.05
	MSML mask DCNN w/four scale layers w/LSI	**44.31**	**42.94**	**20.21**	5.98
	Surf & CNN-based classification model [13]	9.12	7.18	4.59	19.65
	Mask R-CNN [14]	10.91	9.69	7.53	23.91
	Dual-scale CNN-based classification [15] (GoogLeNet & ResNet)	22.81	19.32	10.24	10.98
	CrackPix [16]	25.18	29.33	9.57	14.69

(Best scores shown in bold font).

result in these poor performances (- in Figure 12 (C-3)). In category, the estimated crack-candidates located on the right were eliminated because the Hough transform detected this line. By contrast, the crack candidate located on the left side remained because the Hough transform could not detect it because of the catenary shape of the line. Category includes a large area of several rounded holes that are similar to curved lines. However, a shape made up of several rounded holes can approximate a line and thereby most candidate cracks were effectively eliminated. By contrast, a small area of rounded holes was retained in category because of a resolution problem (Figure S1(c)). A small area of rounded holes can be eliminated like in category if the image is recorded at a closer distance and has adequate resolution. However, the image was recorded far from cracks, and therefore a small area of rounded holes was considered a small line and not eliminated. Image No. 11 also shows low accuracy (M_{rem}: 0.46, M_{elim}: 0.72, and $F1_M$: 0.56).

This image includes the most concerns including many disconnected cracks and rounded holes of TP pixels (Figure 12(d)). This observation indicates that complex facilities in the underground tunnel degrade the accuracy of the proposed method. Thus, automated crack inspection for real-world applications in such tunnels would be extremely difficult and limited. Hence, more efforts should be devoted in future work to increase automated inspection processes. However, the mean values of M_{rem}, M_{elim}, and $F1_M$ are 0.91, 0.71, and 0.76, respectively, for all images from the underground tunnel, suggesting that the proposed method effectively eliminates non-crack features of a line shape even in complex circumstances. It is worth noting that the resolution of images plays an important role in all crack-detection methods. A high-resolution close-up image increases the accuracy of crack detection, which is a reason for most previous studies recording images within a dozen centimeters or using high resolution images [15, 16]. However, this study measured images from 1.4-1.8 m distance because it is a reasonable and economic distance for practical applications considering the camera deployed on the mobile robot. It is also worth noting that the measurements were limited by the camera's unchangeable position. A camera facing cracks exactly in parallel enhances the detection accuracy. However, it is difficult to face all internal structures for recording during a patrol inspection. The resolution and position of the camera are the main reasons for low accuracy in some images.

Regardless of several limitations, the proposed method is more effective than other state-of-the- art crack-detection methods for real-world applications. Table 8 compares the results from the proposed method with those of the other four state-of-the art methods [13–16]. The other four methods were trained with the same public image-sets (described in Table 1) and optimized through the BO method. The results are similar to those from the indoor image-set. However, the estimated APs were higher than those obtained from the indoor image-set. Remarkably, the proposed method outperformed other state-of-the-art crack-detection methods. Specifically, the proposed MSML Mask DCNN is better than other deep learning-based crack-detection methods, suggesting that the proposed deep and wide neural network effectively detects crack features, even though the MSML Mask DCNN was only trained using public datasets. Moreover, the AP of the entire framework combining the MSML Mask DCNN and the LSI shows two times higher accuracy than the others, confirming that false positive pixels are successfully eliminated by the proposed method. In conclusion, a comparison of the proposed method with other four state-of-the-art methods for indoor and tunnel image-sets clearly demonstrates that the proposed method is effective for real-world applications in complex structures with many objects.

7. Conclusions

This study proposes an integrated framework for effective crack detection in images for real-world applications comprising the MSML Mask DCNN and LSI. The proposed method estimates crack candidates using the MSML Mask DCNN only trained by public image-sets based on the principle that cracks have distinct features. Employing the proposed method to test the images demonstrated its effectiveness. Specifically, the MSML Mask DCNN outperformed the state-of the-art neural networks in terms of both accuracy and robustness, whereas the proposed LSI effectively distinguishes non-crack features from cracks. Hence, the proposed method can improve the capability of crack detection on the surface of structures located in complex and various environments. However, the proposed method limits the elimination of other shapes with non-crack features, such as round holes. Future work includes studying a method for eliminating other shapes with non-crack features in crack candidates. Furthermore, a quantitative evaluation of the proposed method should be conducted using more field test images. A novel architecture of a mask neural network should also be studied to enhance the estimation accuracy of crack detection in structures.

Data Availability

The data used in this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Ji-Wan Ham was involved in methodology, formal analysis, writhing the original draft, and tests, Siheon Jeong was involved in data curation, resources, and software, Min--Gwan Kim was involved in annotation of dataset, hard ware, and tests, Joon-Young Park was involved in conceptualization, validation, and visualization, and Ki-Yong Oh was involved in writing, reviewing, editing, supervision, and project administration.

Acknowledgments

This work was supported in part by the Korea Electric Power Corporation through the KEPCO Research Institute under Grant R19TA10 and Korea Institute of Energy Technology Evaluation and Planning (KETEP) grant funded by the Korea government (MOTIE) (20213030020260, Development of Fire detection and protection system for wind turbine).

Supplementary Materials

Figure S1. Partially zoomed-in image of (A) (B-2) in Figure 11, (B) (B-3) in Figure 11, and (C) (C-4) in Figure 12. (*Supplementary Materials*)

References

[1] Y. Yao, S. T. E. Tung, and B. Glisic, "Crack detection and characterization techniques—an overview," *Structural Control and Health Monitoring*, vol. 21, no. 12, pp. 1387–1413, 2014.

[2] M. J. Anitha, R. Hemalatha, and S. Radha, "A survey on crack detection algorithms for concrete structures," *Advances in Smart System Technologies: Select Proceedings of ICFSST 2019*, pp. 639–654, Springer, Berlin, Germany, 2021.

[3] C. Zhang, S. Zhao, Z. Yang, and Y. Chen, "A reliable data-driven state-of-health estimation model for lithium-ion batteries in electric vehicles," *Frontiers in Energy Research*, vol. 10, 2022.

[4] C. Zhang, S. Zhao, and Y. He, "An integrated method of the future capacity and RUL prediction for lithium-ion battery pack," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 3, pp. 2601–2613, 2022.

[5] S. Zhao, C. Zhang, and Y. Wang, "Lithium-ion battery capacity and remaining useful life prediction using board learning system and long short-term memory neural network," *Journal of Energy Storage*, vol. 52, Article ID 104901, 2022.

[6] A. M. A. Talab, Z. Huang, F. Xi, and L. HaiMing, "Detection crack in image using Otsu method and multiple filtering in image processing techniques," *Optik*, vol. 127, no. 3, pp. 1030–1033, 2016.

[7] V. Maksimovic, M. Petrovic, D. Savic, B. Jaksic, and P. Spalevic, "New approach of estimating edge detection threshold and application of adaptive detector depending on image complexity," *Optik*, vol. 238, Article ID 166476, 2021.

[8] P. Wang and H. Huang, "Comparison analysis on present image-based crack detection methods in concrete structures," in *Proceedings of the 2010 3rd international congress on image and signal processing Yantai, China*, vol. 5, pp. 2530–2533, Yantai, China, October 2010.

[9] J. Dai, K. He, and J. Sun, "Convolutional feature masking for joint object and stuff segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3992–4000, Boston, MA, USA, June 2015.

[10] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "DeepCrack: learning hierarchical convolutional features for crack detection," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1498–1512, 2019.

[11] M. R. Saleem, J. W. Park, J. H. Lee, H. J. Jung, and M. Z. Sarwar, "Instant bridge visual inspection using an unmanned aerial vehicle by image capturing and geo-tagging system and deep convolutional neural network," *Structural Health Monitoring*, vol. 20, no. 4, pp. 1760–1777, 2020.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, Cham, Switzerland, October 2015.

[13] H. Kim, E. Ahn, M. Shin, and S. H. Sim, "Crack and noncrack classification from concrete surface images using machine learning," *Structural Health Monitoring*, vol. 18, no. 3, pp. 725–738, 2019.

[14] B. Kim and S. Cho, "Automated multiple concrete damage detection using instance segmentation deep learning model," *Applied Sciences*, vol. 10, no. 22, p. 8008, 2020.

[15] F. Ni, J. Zhang, and Z. Chen, "Zernike-moment measurement of thin-crack width in images enabled by dual-scale deep learning," *Computer-Aided Civil and Infrastructure Engineering*, vol. 34, no. 5, pp. 367–384, 2019.

[16] M. Alipour, D. K. Harris, and G. R. Miller, "Robust pixel-level crack detection using deep fully convolutional neural networks," *Journal of Computing in Civil Engineering*, vol. 33, no. 6, Article ID 04019040, 2019.

[17] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, and M. Jagersand, "U2-Net: going deeper with nested U-structure for salient object detection," *Pattern Recognition*, vol. 106, Article ID 107404, 2020.

[18] D. Kim, S. Kim, S. Jeong et al., "Rotational multipyramid network with bounding-box transformation for object detection," *International Journal of Intelligent Systems*, vol. 36, no. 9, pp. 5307–5338, 2021.

[19] V. Badrinarayanan, A. Kendall, and R. S. Cipolla, "SegNet: a deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Las Vegas, NV, USA, June 2017.

[21] Q. Zhao, T. Sheng, Y. Wang et al., "M2Det: a single-shot object detector based on multi-level feature pyramid network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 9259–9266, Honolulu, HI, USA, July 2019.

[22] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proceedings of the International Conference on Learning Representations 2014*, Banff, Canada, April 2014.

[23] W. Abu-Ain, S. N. H. S. Abdullah, B. Bataineh, T. Abu-Ain, and K. Omar, "Skeletonization algorithm for binary images," *Procedia Technology*, vol. 11, pp. 704–709, 2013.

[24] R. Lerallut, É. Decencière, and F. Meyer, "Image filtering using morphological amoebas," *Image and Vision Computing*, vol. 25, no. 4, pp. 395–404, 2007.

[25] R. M. Haralock and L. G. Shapiro, "Mathematical morphology," in *Computer and Robot Vision*, pp. 174–185, Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA, 1991.

[26] J. Malik, R. Dahiya, and G. Sainarayanan, "Harris operator corner detection using sliding window method," *International Journal of Computer Application*, vol. 22, no. 1, pp. 28–37, 2011.

[27] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[28] K. L. Wu and M. S. Yang, "Mean shift-based clustering," *Pattern Recognition*, vol. 40, no. 11, pp. 3035–3052, 2007.

[29] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[30] T. W. Ke, J. J. Hwang, Z. Liu, and S. X. Yu, "Adaptive affinity fields for semantic segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 587–602, Munich, Germany, September 2018.

[31] V. Wegmayr, A. Sahin, B. Saemundsson, and J. Buhmann, "Instance segmentation for the quantification of microplastic fiber images," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2210–2217, Snowmass, CO, USA, March 2020.

[32] J. Fan, Z. Zhang, C. Song, and T. Tan, "Learning integral objects with intra-class discriminator for weakly-supervised semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4283–4292, Seattle, WA, USA, June 2020.

[33] J. König, M. D. Jenkins, M. Mannion, P. Barrie, and G. Morison, "Optimized deep encoderdecoder methods for crack segmentation," *Digital Signal Processing*, vol. 108, Article ID 102907, 2021.

[34] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, "Feature pyramid and hierarchical boosting network for pavement crack detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 4, pp. 1525–1535, 2020.

[35] M. Abdellatif, H. Peel, A. G. Cohn, and R. Fuentes, "Combining block-based and pixelbased approaches to improve crack detection and localisation," *Automation in Construction*, vol. 122, Article ID 103492, 2021.

[36] Q. Li, Q. Zou, D. Zhang, and Q. Mao, "FoSA: F∗ Seed-growing Approach for crack-line detection from pavement images," *Image and Vision Computing*, vol. 29, no. 12, pp. 861–872, 2011.

[37] M. Minagawa, Y. Matsuda, Y. Fujiwara, and Y. Fujii, "Relationship between crack propagation and the stress intensity factor in cutting parallel to the grain of hinoki (Chamaecyparis obtusa)," *Journal of Wood Science*, vol. 64, no. 6, pp. 758–766, 2018.

[38] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, pp. 1558–1570, 2015.

[39] Y. Bai, H. Sezen, and A. Yilmaz, "End-to-end deep learning methods for automated damage detection in extreme events at various scales," in *Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6640–6647, Milan, Italy, Janu 2021.

[40] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, 2018.

[41] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. O. Koyama, "A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, Anchorage, AK, USA, July 2019.

[42] D. Bonet, A. Ortega, J. Ruiz-Hidalgo, and S. Shekkizhar, "Channel-wise early stopping without a validation set via NNK polytope interpolation," in *Proceedings of the 2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pp. 351–358, Tokyo, Japan, December 2021.

[43] C. Corneanu, M. Madadi, S. Escalera, and A. Martinez, "Explainable early stopping for action unit recognition," in *Proceedings of the 2020 15th IEEE international conference on automatic face and gesture recognition*, pp. 693–699, Buenos Aires, Argentina, November 2020.

[44] Y. Chai, H. Liu, and J. Xu, "Glaucoma diagnosis based on both hidden features and domain knowledge through deep learning models," *Knowledge-Based Systems*, vol. 161, pp. 147–156, 2018.

[45] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2014.

[46] Z. Zhang, "Improved Adam optimizer for deep neural networks," in *Proceedings of the 2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pp. 1-2, Banff, Canada, June 2018.

[47] T. P. Quinn, V. Le, and A. P. Cardilini, "Test set verification is an essential step in model building," *Methods in Ecology and Evolution*, vol. 12, no. 1, pp. 127–129, 2021.

[48] O. Bailo, F. Rameau, K. Joo, J. Park, O. Bogdan, and I. S. Kweon, "Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution," *Pattern Recognition Letters*, vol. 106, pp. 53–60, 2018.