Construction and Building Materials 359 (2022) 129438

Contents lists available at ScienceDirect

Construction and Building Materials

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

Deep learning-based masonry crack segmentation and real-life crack length measurement

L. Minh Dang [b,c,1], Hanxiang Wang [a], Yanfen Li [a,1], Le Quan Nguyen [a], Tan N. Nguyen [d], Hyoung-Kyu Song [b], Hyeonjoon Moon [a,*]

[a] Department of Computer Science and Engineering, Sejong University, 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea
[b] Department of Information and Communication Engineering, and Convergence Engineering for Intelligent Drone, Sejong University, Seoul, Republic of Korea
[c] Department of Information Technology, FPT University, Ho Chi Minh city 70000, Viet Nam
[d] Department of Architectural Engineering, Sejong University, 209 Neungdong-ro, Gwangjin-gu, 05006, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Masonry building
Crack segmentation
Deep learning
Measurement
Image processing

ABSTRACT

While there have been a considerable number of studies on computer vision (CV)-based crack detection on concrete/asphalt public facilities, such as sewers and tunnels, masonry-related structures have received less attention. This research seeks to implement an automated crack segmentation and a real-life crack length measurement of masonry walls using CV techniques and deep learning. The main contributions include (1) a large dataset of manually labelled images about various types of Korea masonry walls; (2) a careful performance evaluation of various deep learning-based crack segmentation models, including U-Net, DeepLabV3+, and FPN; and (3) a novel algorithm to extract real-life crack length measurement by detecting the brick units. The experimental results showed that deep learning-based masonry crack segmentation performed significantly better than previous approaches and could provide a real-life crack measurement. Therefore, it has a huge potential for motivating masonry-based structure investigation.

1. Introduction

Masonry is a matured and well-established construction technique commonly used in historic and modern architectures. There has also been growing interest in implementing masonry for future infrastructure [1]. A masonry building includes various masonry materials, such as bricks, ashlars, blocks, and stones, usually stuck together using mortar. Masonry architecture is well-known for its long service life due to its capability to be incrementally fixed [2]. Moreover, the modular nature of masonry (individual units and sacrificial mortar) is the main reason it is cheaper to maintain than concrete slabs.

During its lifetime, structural damage to masonry can occur for a variety of reasons. Cracks can potentially appear due to environmental factors, such as inconsistent material adjacency, earthquakes, thermal stress, hygroscopic stress, and mechanical stress [3,4]. To date, the maintenance of the masonry architecture and structural damage (i.e., cracks, spalls, etc.) has been mainly conducted via the following techniques: ground-level inspection [5,6], tactile inspections (cherry pickers, scaffolding, or ropes) [7], and UAV-based visual inspection [8]. Most techniques except those that are UAV-based are performed manually, which is labor-intensive, time-consuming, and error-prone. Even the modern method of using drones to capture images can be laborious and subjective because after the images/videos are captured, an inspector is required to investigate them to manually validate the structure's condition.

Earlier studies have proved that conventional image processing and CV techniques can be applied to reconstruct digital documentaries of masonry architectures, which can be used to detect the shape and position of a masonry unit or perform crack segmentation [9]. Moreover, the feature extraction process can produce the numerical model of masonry for more detailed analysis. It usually contains three main processes [10], which include 1) image preprocessing to reduce noise introduced during the data collection, 2) manual feature detection to extract essential features from the collected dataset, and 3) segmentation techniques to partition the images into distinctive regions. However, conventional approaches showed poor performance when facing unseen data, cost more time due to the manual feature extraction process, and were sensitive to digital noise generated by uneven lighting, texture, and

* Corresponding author.
E-mail address: hmoon@sejong.ac.kr (H. Moon).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.conbuildmat.2022.129438
Received 6 July 2022; Received in revised form 9 October 2022; Accepted 10 October 2022

color [11].

During recent decades, deep learning has shown remarkable performance in standard CV topics, which include classification [6,12], object detection [13], and segmentation [5,10]. Moreover, deep learning models are semantically robust because they can learn abstract and coarse features automatically from the raw data [14]. Fully convolutional networks (FCN) [15] are considered a well-known deep learning-based image segmentation model because they can accept images at any resolution as the input by transforming fully-connected layers of a CNN structure into convolutional layers.

Various FCN structures can be potentially applied to perform masonry crack segmentation. First, U-Net [16], which follows the FCN architecture, was initially introduced for segmenting biomedical images. U-Net shows that acquiring the spatial features through the transposed convolution concept is feasible. U-Net is a standard segmentation model and has been implemented in various CV applications, especially for relatively small datasets. The second SOTA FCN structure proposed for general semantic segmentation is DeepLabV3+ [17], which has simpler structures and can offer cheaper computing power while delivering better performance. Even though FCN architecture demands a huge and carefully annotated dataset in order to provide good performance, it has the potential to obtain representative features by providing a training dataset that covers a large collection of various cases. In addition, for training the model on small datasets, the transfer learning technique, which relies on a pre-trained CNN model on other benchmark datasets as a starting point, can be applied to enable the model to learn a new task easier and faster [18]. Therefore, transfer learning is encouraged and can be applied to any FCN architecture.

Deep learning models have already been implemented to perform some structural engineering tasks due to their high prospect of assisting structural inspection and monitoring applications. In the case of masonry crack investigation, deep learning offers the solutions to classify, detect, and locate segmented cracks. For instance, Hallee et al. trained a customized CNN model on a masonry dataset collected in the laboratory, containing 5 distinctive classes (Cracked, uncracked, vague, partial, and no-bricks) [19]. The authors showed that the CNN model achieved higher performance than ML models trained on handcrafted features with the highest accuracy of 81 %. However, the model performance declined sharply when it was tested on real-world images. The same year, Dais et al. [6] investigated various deep learning models for patch classification and semantic segmentation tasks for masonry cracks. The experimental results showed that fine-tuned MobileNet, which was pretrained on the ImageNet dataset, obtained the highest crack classification accuracy of 95 %. For the crack segmentation task, U-Net model with the encoder path replaced by pretrained MobileNet delivered the highest F1 score of 79 %. The authors also explained the importance of transfer learning based on a pretrained backbone, which improved the F1 score to about 4 %. Valero et al. [12] extracted a 3D point cloud-based statistical representation of data to train a logistic regression algorithm to perform the classification and detection of chromatic alterations and defects on ashlar masonry walls.

Even though many studies have been proposed to perform crack segmentation for public facilities, such as sewers [20] and tunnels [9,10], less attention has been given to masonry crack segmentation research. Moreover, existing implementations of CNN models for masonry crack investigation mostly use images from designated sites for training and testing, leading to high performance in a strictly controlled environment [2,12,19]. Furthermore, some particular studies [5,12] trained the models by assuming a typical masonry unit size to develop a sliding window that excludes joints, which is a challenge for use in real-life applications. Finally, few studies have been working on recognizing practical data of the segmented masonry cracks (length, thickness). This information is a critical factor affecting the masonry investigation and maintenance processes.

As a result, this study couples crack segmentation and brick detection techniques and then automatically provides real-life length measurements of the detected cracks. Crack segmentation was performed using SOTA FCN architectures, whereas brick detection was done using the Mask-RCNN model. The presented work is essential for automatic documentation, assessment, and maintenance of existing masonry structures from digital images.

2. Dataset

2.1. Data collection

Initially, a dataset was collected containing regular masonry structure images in Korea without the rubble patterns. Most of the images were downloaded from the internet (Naver, Google search), while a small part of the dataset was collected in different regions of Korea using different devices under different resolutions (drones and smartphones). A generalization of the dataset is also considered by selecting images with cracks, doors, and windows. Those images contain various masonry units, color, illumination, and capture-angle. In total, 200 images with the size ranging from 780 × 355 to 2880 × 1920 were collected at the end of the data collection process.

2.2. Crack segmentation dataset

Fig. 1 provides a detailed explanation for making the masonry crack segmentation dataset from the original raw images. First, the labelling process was carried out using the original dataset (Fig. 1(a)). Three experts from a masonry investigation company[2] were given a one-month labelling task. On average, five images were labelled per person per day. The labelling process began with an automated Python-based annotation application, which implements standard processing techniques, such as morphological operations and edge detection, to automatically recognize and highlight the cracks in red. Next, the experts reviewed the outputs of the application and corrected them if necessary. This step generated 500 crack-annotated images based on the original 500 raw images. Secondly, a color-based thresholding technique was implemented to transform the crack-highlighted images into a binarized image with the crack's regions in white and the other areas in black, as displayed in Fig. 1(b). The thresholding process also reduces the color channel impact during the training process. Finally, a patch extraction process was applied to the original images and their corresponding binary images because the sizes of those images varied. The sliding

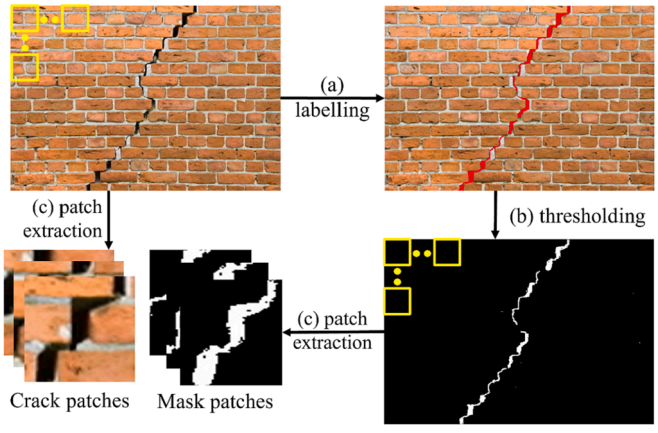

Fig. 1. Step-by-step process of the masonry crack data generation. Note: Three primary processes are (a) labelling (confirmed by experts), (b) thresholding, and (c) patch extraction.

[2] https://deepinspection.ai.

window technique with a size of 224 × 224 and a stride of 0.1 was utilized to convolve over the raw images and their corresponding binary images simultaneously to acquire a list of crack and mask patches (Fig. 1 (c)). The patches without cracks were automatically removed.

After performing all three steps, a total of 2670 crack patches and 2670 mask patches were obtained. The segmentation data was then split into training and testing datasets using the following holdout strategy. 80 % of the original dataset was randomly selected as the training data, which was then divided into the training data for training the model and validation data for fine-tuning the hyperparameters. The remaining 20 % of the original dataset was selected as the testing data to evaluate the framework's performance. A detailed description for training, validation, and testing datasets is provided in Table 1.

2.3. Brick detection dataset

The direction a masonry brick unit is put has a huge impact on how it appeals to aesthetics, supports the structure, and withstands wear and tear. For the collected masonry dataset, the brick was primarily laid in two orientations, as depicted in Fig. 2. The most common forms are *stretcher*, which lays the bricks horizontally in a wall with the long edge parallel to the surface. Another approach is *header,* which lays the bricks horizontally in a wall with the shorter end exposed to the surface.

Different from the crack segmentation dataset, the annotation tool used for brick detection is a well-known open-source Python application named labelImg[3] based on Qt5. The dataset was then split into training, validation, and testing sets using the same holdout strategy. The number of images is displayed in Table 1, the brick detection dataset column.

3. Methodology

Fig. 3 depicts the overall procedure of the masonry crack segmentation and measurement framework, which contains three primary processes.

Data preparation, described in Section 2, is the starting point of the framework. The masonry crack segmentation and the brick detection datasets are created at the end of the data preparation step. Two parallel processes are then carried out for two separate tasks, crack segmentation, and brick detection. For the crack segmentation task, various deep learning models, including FCN, U-Net, and DeepLabV3+, were implemented, and their performance was tested carefully to discover the best model for the collected masonry crack segmentation dataset. Several post-processing techniques were applied to remove noise from the segmented outputs. A skeletonization process was then carried out to extract the crack skeleton. On the other hand, for the brick detection task, a Mask-RCNN model was trained to detect the brick units from the input image, which is required to compute the brick's width in order to obtain the measurement in real-life in millimeters (mm). Finally, the extracted crack skeleton and real-life measurements are used to calculate the crack length and thickness.

Table 1
Number of images for the proposed masonry crack segmentation and brick detection dataset.

	Crack segmentation		Brick detection	
	Crack patches	Mask patches	Original images	Annotated images
Training	1708	1708	256	256
Validation	428	428	64	64
Testing	534	534	80	80
Total	2670	2670	400	400

[3] https://github.com/tzutalin/labelImg.

The masonry crack segmentation model and post-processing are explained in Section 3.1. The brick detection and real-life measurement extraction are then discussed in Section 3.2. Finally, Section 3.3 describes the real-life crack length extraction process.

3.1. Crack segmentation and post-processing

3.1.1. Crack segmentation

In this study, the segmentation models used for training the masonry crack segmentation are U-Net, FCN, and DeepLabV3 +. Transfer learning was implemented to train the models using the created dataset. In addition, various backbones, including ResNet-50, VGG-19, InceptionV4, MobileNetV3, and Xception, were examined to find the model's most suitable backbone.

The loss function's main objective is to reduce the error during training and optimize the weights to minimize the loss during the subsequent evaluation. Two different loss functions, including Binary-Cross-Entropy (BCE) and Focal-Loss (FL), were considered in this study in order to determine the most appropriate loss function for each model. All experiments were based on the Adam optimizer with a learning rate of 1E-4 and decay set to 1E-6.

3.1.2. Post-processing

The segmented outputs can contain broken cracks separated from a full crack or various types of noise. As a result, a post-processing process, which tries to connect broken cracks and reduce noise using morphological operations, was implemented to improve the predicted outputs.

Firstly, the closing operation was applied to connect broken cracks. It is described as follows:

$$A \cdot B = (A \oplus B) \ominus \mathrm{B} \tag{1}$$

where $\oplus$ is the dilation process, which adds the pixels, while $\ominus$ depicts the erosion process that narrows the pixels of the boundary of the object. The dilation computes a local maximum, while erosion computes a local minimum over an area of a kernel B.

After connecting the broken cracks using the closing operation, the remaining noise was further reduced by performing the opening process, as illustrated in Equation (2). It is contrary to the closing process, where the dilation process was implemented after the erosion process.

$$A^{\circ}B = (A \ominus \mathrm{B}) \oplus B \tag{2}$$

This research applied a 3 by 3 point kernel to remove noise from the output masks and obtain only crack pixels.

3.1.3. Skeletonization

Skeletonization is usually applied to find the single-pixel skeleton of an object that has a detected masonry crack. Skeletonization accurately extracts the crack's topology, which can be used to provide a detailed analysis of the crack. Previously, different methods have been introduced to generate the object's skeleton accurately, such as digital patterns thinning [21], 3D medial surface thinning [22], morphological thinning [23], and medial axis [24].

Fig. 4 shows the crack topology output of the four skeletonization methods for two cases, including *Case 1* (simple crack) and *Case 2* (complicated crack). Overall, the output crack topologies were captured correctly for all approaches except the morphological thinning (4th column), which produced a broken output for *Case 1* and wrongly generated various additional crack branches for *Case 2*.

Each skeletonization method has strengths and weaknesses, so the researcher must assess them carefully and choose the most suitable method. For example, 3D medial surface thinning [22] and digital patterns thinning [21] require a high computational cost for *Case 2* because the iterative processes grow exponentially for complicated cracks, so they are suitable for applications that demand precise extraction with longer processing times. The medial axis [24] is used in this study

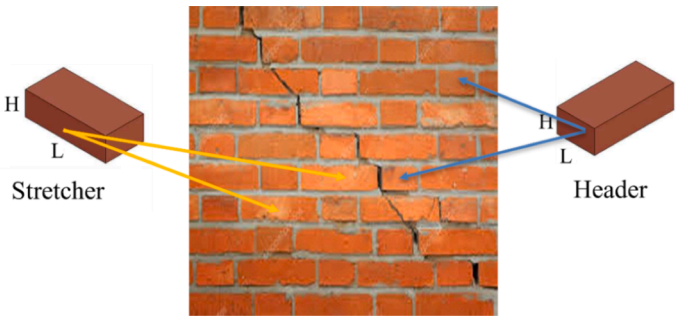

Fig. 2. Description of the possible orientations of the masonry brick.

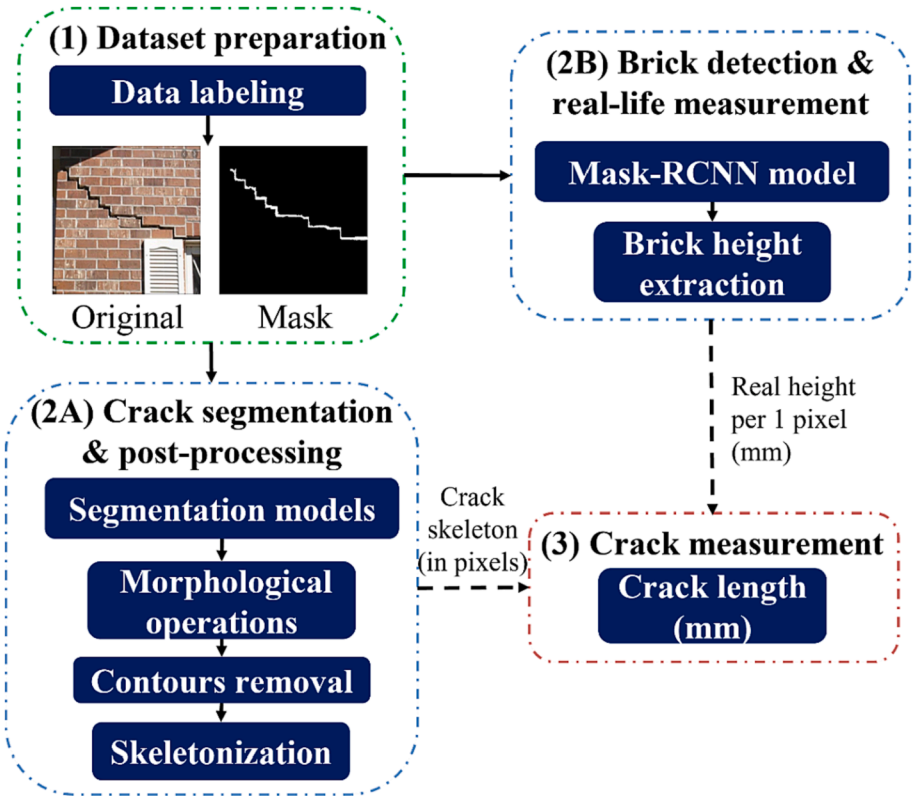

Fig. 3. Detailed explanation of the masonry crack segmentation and measurement framework. Note: Three primary tasks are (1) data preparation, (2A) crack segmentation and post-processing, (2B) brick detection and measurement, and (3) crack measurement.

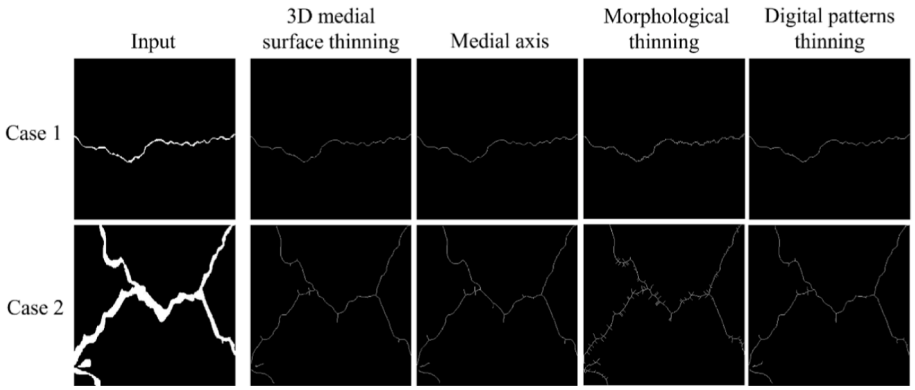

Fig. 4. Visualization of the four different skeletonization approaches for two different cases of cracks.

because it produces accurate crack topology for cracks with fewer branches (same as masonry cracks) and requires less computing power than other algorithms.

3.2. Brick detection and real measurement extraction

It is feasible to provide the actual measurement of a crack in mm due to the masonry brick units' fixed dimensionality and ratio nature. This section focuses on detecting the bricks and evaluating the detected brick in order to provide the actual measurement per 1 pixel to facilitate the measurements of the segmented cracks in real-world data.

3.2.1. Brick detection

The mask region-based convolutional neural network (Mask RCNN) is a state-of-the-art image segmentation deep learning model [25]. It is an extension of Faster RCNN, where an additional branch is added for predicting the object mask of each candidate object alongside the exiting two branches for the bounding box and class label. The Mask RCNN network implemented in this study has three main stages, as displayed in Fig. 5.

First, for the backbone part, pre-trained ResNet50/101 models on the ImageNet dataset [26] were used to extract low-level features from training brick images. The backbone also includes a feature pyramid network (FPN) to enable the representation of the brick target on multiple scales. FPN is also effective for detecting small targets like brick units. An up-sampling process was implemented to combine the FPN's top-level features and the extracted features, each layer producing its specific feature maps.

Second, the output feature maps produced by the backbone were fed into a region proposal network (RPN). A brick target is usually a small object with varying sizes due to the capturing position. Consequently, three area-scale anchors were proposed (16×16, 32×32, and 64×64) based on the possible number of brick pixels an image may contain. The rectangular box's length to width ratio in an input image was about two, so three anchors (1:1, 2:1, and 1:2) were selected. RPN uses a cross-entropy loss to learn and verify the created anchors and SmoothL1 loss to adjust the anchors' coordinates. The output of the RPN is a collection of ROIs that possibly contain brick units.

The next step is ROI alignment, which aims to map the extracted ROIs to obtain the target features using bilinear interpolation to satisfy the input demands of the FC.

Finally, the outputs of the ROI alignment were fed into three different output paths, including the regression layer for bounding box prediction, the FC layer for class prediction, and the FCN for generating the target mask.

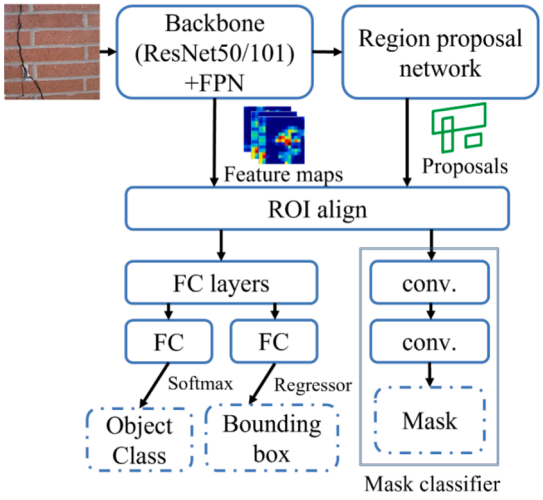

Fig. 5. Mask-RCNN architecture for the masonry brick unit detection task.

3.2.2. Real-life measurement extraction

After the bricks were detected using the Mask-RCNN model, various processes were performed with the primary goal of providing the real measurement in mm. Fig. 6 represents the detailed description of the brick measurement process proposed in this study.

The length (L) and height (H) of all detected bricks were first computed based on the bounding box information and added to a matrix. Although there are two main ways the bricks were laid, including stretcher and header, as described in Section 2.3, there remain other cases of the detected brick, such as brick obscured by object and broken brick, etc., which resemble the header-based laid brick. Consequently, it is unavoidable that the computed L and H varied among the detected bricks.

A K-mean clustering algorithm is implemented in order to cluster the computed length and height data for each detected brick with a presumed k value of 2 (stretcher and header). K-means clustering belongs to a matrix factorization issue, so the data need to be transformed into a $L \times H$ matrix.

The L/H ratio for each centroid of each cluster was calculated to filter out the cluster that has the highest L/H ratio, which indicates the stretcher-type bricks.

Each L/H ratio of the brick in the stretcher-type cluster was then compared with the available standard L/H ratios, which are *190/90*, *205/90*, and *230/90,* according to the Korea Building Code Draft (KBC-205 0602.2) [27]. Bricks that match the standard L/H ratios (with an error of less than 0.1) were added to a list. There are two possible cases:

The list is not empty: the average H of all the bricks in the list was computed and used to get the real mm per 1px based on the matched standard L/H ratio.

The list is empty: a perspective transformation method is carried out on the image based on extracting the position (x, y) of the four positions of a brick (top left, top right, bottom left, and bottom right). Step i was then performed again.

3.3. Crack measurement

Previous studies have proved that the length of a crack can be computed based on the extracted crack skeletons [28,29]. However, the output results were in pixels, which could not be applied to real-life applications. By automatically extracting the actual measurement of mm per pixel in the previous section, it is feasible to provide the actual measurement in real-life for all segmented cracks. Those measurements provide experts with more analytical tools to determine the masonry architecture's condition.

The equation for computing the length (L) of a crack is described as follows:

$$L = \int_c \delta dl \cong \sum \delta dl \tag{3}$$

where δ depicts the geometric distortion calibration and dl is the finite length of L. Initially, δ is used as a calibration value for possible pixel displacements in the predicted mask images. The geometric distortion was solved by performing the perspective transformation in Section 3.2.1. As a result, δ was set to 1, which allows L to be calculated by directly counting the generated skeleton pixels.

4. Experimental results

In this work, all the experiments were implemented on an Ubuntu 18.04 machine equipped with an RTX R5000 24 GB GPU and 252 GB of DDR4 RAM.

The main metrics computed to evaluate the model's performance and robustness are explained in Section 4.1. Section 4.2, including various experiments to verify the proposed masonry crack segmentation results. Then, the model used in this study was compared with the other

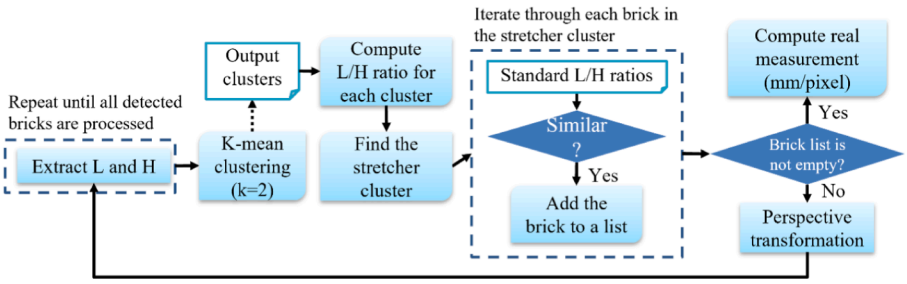

Fig. 6. Detailed description of the real-life measurement extraction process.

state-of-the-art crack segmentation models to demonstrate its superiority. In Section 4.3, in order to facilitate the real-life crack length measurement process, the performance of the Mask-RCNN-based brick detection model is described. After that, the performance of the post-processing process and skeletonization are shown in Section 4.4 and Section 4.5. Finally, Section 4.6 provides a detailed experiment for real-life masonry crack measurement based on the segmentation and brick measurement results.

4.1. Evaluation metrics

Masonry crack segmentation belongs to the binary classification issue because each pixel in the input image is classified into either crack (white) or background (black). As a result, intersection over union (IoU) is calculated using the four components, including true positive (TP), true positive (TN), false positive (FP) and false negative (FN), from a binary confusion matrix. IoU computes the overlap between the ground truth mask and the prediction mask. It is formulated as follows:

$$IoU = \frac{TP}{TP + FN + FP} \tag{4}$$

4.2. Masonry crack segmentation model

Multiple versions of segmentation models were implemented to determine the optimal combination of backbone, parameters, and loss function that form the best segmentation model. All models were tested using an Adam optimizer, with a learning rate of 1E-4 over 30 epochs.

Table 2 shows that some versions of the three models delivered high validation IoU, accuracy, precision, recall, and F1. For each model, the optimal parameters were as follows:

- U-Net (#4): pretrained ResNet-50 backbone on ImageNet dataset, "sigmoid" activation function, focal loss, and decoder channels of 256, 128, 64, 32, and 16.
- FPN (#3): pretrained VGG19 backbone on ImageNet dataset, "sigmoid" activation function, BCE loss, decoder pyramid channels of 512, decoder segmentation channels of 128, and dropout of 0.2.
- DeeplabV3+ (#1): pretrained ResNet-50 backbone on ImageNet dataset, "Sigmoid" activation, BCE loss, and decoder channels of 256.
- DeepLabV3+ (#1), which used the ResNet-50 backbone and BCE loss, achieved the highest accuracy and IoU of 98 % and 0.97, respectively (Table 2), on the training dataset. As a result, it was selected as the default model for performing masonry crack segmentation.

Fig. 7 demonstrates the training/validation losses and accuracies of the DeepLabV3+ (#1) model. The accuracy of the training and validation processes increases significantly to above 95 %, while the training and validation losses decrease remarkably to under 0.15 after epoch 3. The validation accuracy and loss increase slowly after epoch 3 and stabilize before reaching the highest validation accuracy of 97 % and validation loss of 0.12 at epoch 40.

After selecting DeepLabV3+'s as the primary model used for crack segmentation, its robustness was further investigated by evaluating it along with two existing state-of-the-art segmentation models: DeepCrack [31] and CrackNet [32]. The two models were implemented with hyperparameters that strictly follow those defined in the reference papers.

As shown in Table 3, accuracy and IoU scores were computed and

Table 2
Detailed description for various parameters implemented for each provided model.

Model	Backbone	Loss function	Index	Loss	Accuracy	Precision	Recall	F1	IoU
U-Net [16]	–	BCE	#1	0.31	95	92	98	95	0.91
		Focal loss	#2	**0.15**	94	92	**99**	94	0.91
	ResNet-50	BCE	#3	0.23	94	95	98	**97**	0.94
		Focal loss	**#4**	**0.15**	**96**	**96**	98	**97**	**0.95**
	VGG19	BCE	#5	0.24	95	94	96	96	0.92
		Focal loss	#6	**0.15**	95	**96**	**99**	**97**	**0.95**
	InceptionV4	BCE	#7	0.29	92	93	95	95	0.92
		Focal loss	#8	0.17	88	**96**	91	93	0.87
FPN [30]	ResNet-50	BCE	#1	0.18	93	91	93	92	0.85
		Focal loss	#2	0.23	67	95	69	79	0.67
	VGG19	BCE	**#3**	0.15	**94**	**98**	95	**97**	**0.94**
		Focal loss	#4	**0.12**	90	**98**	92	95	0.9
	InceptionV4	BCE	#5	0.22	92	95	**96**	95	0.92
		Focal loss	#6	0.27	88	**98**	89	93	0.88
DeeplabV3+ [17]	ResNet-50	BCE	**#1**	**0.02**	**98**	97	**99**	**98**	**0.97**
		Focal loss	#2	0.07	94	96	98	97	0.94
	MobileNetV3	BCE	#3	0.09	**97**	97	**99**	**98**	**0.97**
		Focal loss	#4	**0.02**	95	97	97	97	0.95
	Xception	BCE	#5	0.24	93	**98**	94	96	0.93
		Focal loss	#6	0.03	95	95	**99**	97	0.95

Note. Bold numbers indicate the highest result for each model. The yellow color shows the best overall performance of each model.

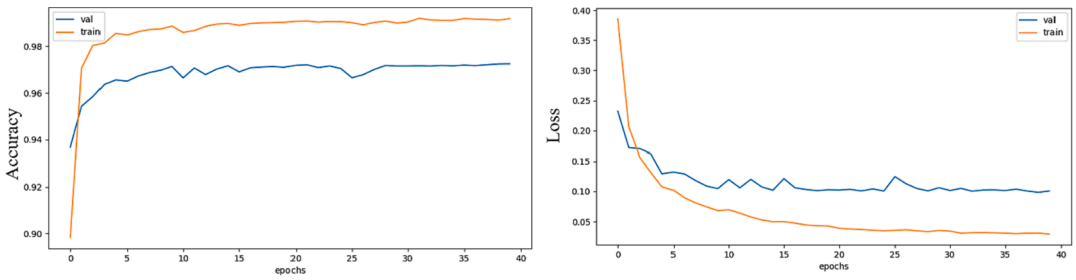

Fig. 7. DLV3 + model performance on the crack segmentation validation datasets.

Table 3
Proposed model's performance compared to two other deep learning models.

Model	Accuracy	IoU
CrackNet [32]	87	0.86
DeepCrack [31]	94	0.92
DeepLabV3+ (Our)	98	0.97

compared between the three crack segmentation models. Generally, they demonstrated high accuracy and IoU values of over 85 % and 0.85, respectively. The proposed masonry crack segmentation based on the DeepLabV3 + model achieved the highest IoU of 0.97. The DeepCrack model came in second place with an IoU of 0.92. Finally, the CrackNet model displayed the lowest IoU of 0.86 and an accuracy of 87 %. As a result, the DeepLabV3 + outperformed the previous crack segmentation model on the proposed masonry crack segmentation dataset.

Fig. 8 demonstrates the model predicted outputs for three different scenarios of the input image. For each scenario, the first column is the original image, the second column indicates the ground truth mask, where the actual cracks are highlighted, and the final column shows the predicted masks that were the output of the DeepLabV3 + model. Generally, the DeepLabV3 + model localized the cracks correctly and robustly for different conditions. For example, the model precisely predicted the crack mask for the input image for simple cracks on the masonry surface as in the first scenario. It also correctly localized the cracks even when the input was slightly skewed, as in the second scenario. It is even more challenging in the last case. Even though some parts of the window resemble the cracks, the model still robustly localized the cracks. Moreover, the proposed model greatly lowered the number of disjoint cracks from the output mask by joining them using the post-processing process.

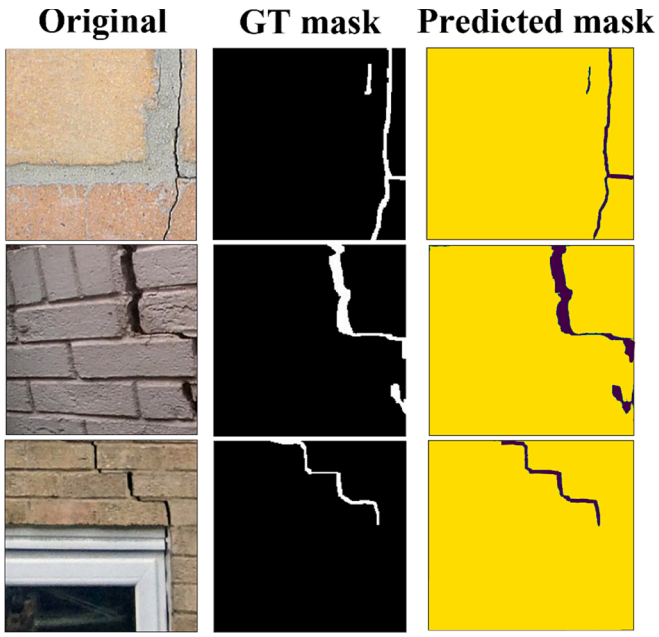

Fig. 8. Comparison between the predicted masks using the proposed DeepLabV3 + model and the ground truth masks on various scenarios of the input images.

4.3. Brick detection

Fig. 9 shows the bounding box loss and mask loss graph for the brick detection process using the Mask-RCNN framework. Both the bounding box loss and the mask loss declined considerably and stopped at 0.16 and 0.1, respectively, at the end of epoch 20.

Fig. 10(b) displays the brick unit detection result using the trained Mask-RCNN model for two input images. It can be seen that the detection rate was high because the model could detect the two types of brick (stretcher and header) as well as bricks obscured by objects (the first input image). As shown in Fig. 10(a), a bounding box and a brick mask were predicted for each detected brick.

The detected bricks were then fed into the brick unit measurement algorithm explained in Section 3.2.1 in order to recognize the actual mm per pixel result.

4.4. Post-processing process

Post-processing is a crucial process that can reduce noise and missing cracks and connect nearby cracks to create a uniform crack. Fig. 11 shows a sample of the effectiveness of the post-processing method. In the case of no post-processing process, there is a crack in the upper right part of the image, but the model segmented it too narrowly, leading to the missing crack in the skeletonization result. However, as displayed in the bottom left image, the mentioned crack region was enhanced when morphological operations were implemented. As a result, the extracted crack skeleton was correct, which contained all the crack regions. In addition, potential noise from the predicted image can also be effectively removed after applying the post-processing method.

4.5. Skeletonization

After localizing the masonry cracks, it is essential to automatically compute the crack length in order to provide additional statistical information to the inspectors to make a more precise decision. The automated crack length measurement algorithm proposed by this study was based on the crack skeletons explained in Section 3.2. The skeletonization process converts the segmented cracks into a single-pixel crack skeleton. The median-axis skeletonization method was applied to extract the skeleton of each segmented crack.

Fig. 12 displays the skeletonization output for two input masonry crack images. The first column contains the input RGB images, the second column is the segmented outputs, and the last column displays the generated skeletons for the segmented images. The proposed model accurately segmented masonry cracks, even with challenging input. Moreover, the medial axis approach precisely produced the

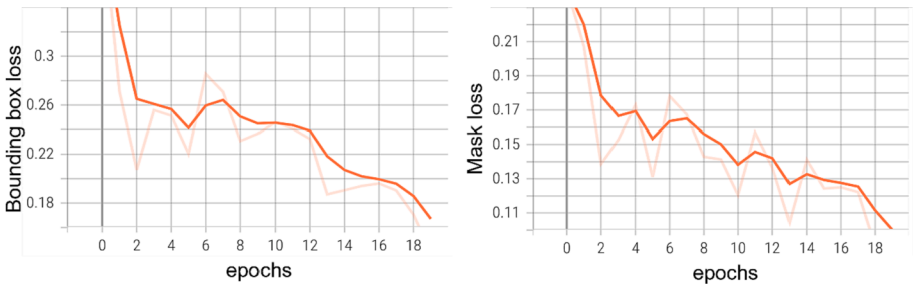

Fig. 9. Mask-RCNN's performance on the brick detection validation set.

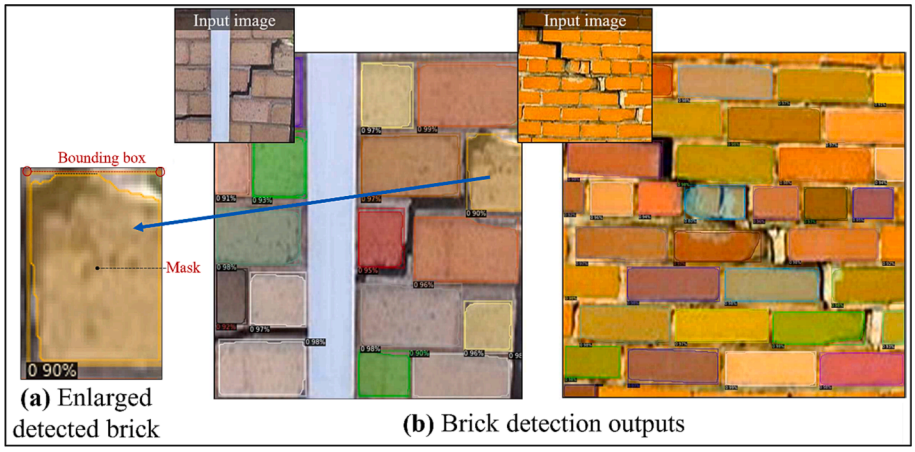

Fig. 10. Detected bricks on different types of masonry brick units using the Mask-RCNN model. a) enlarged version of a detected brick; b) brick detection results.

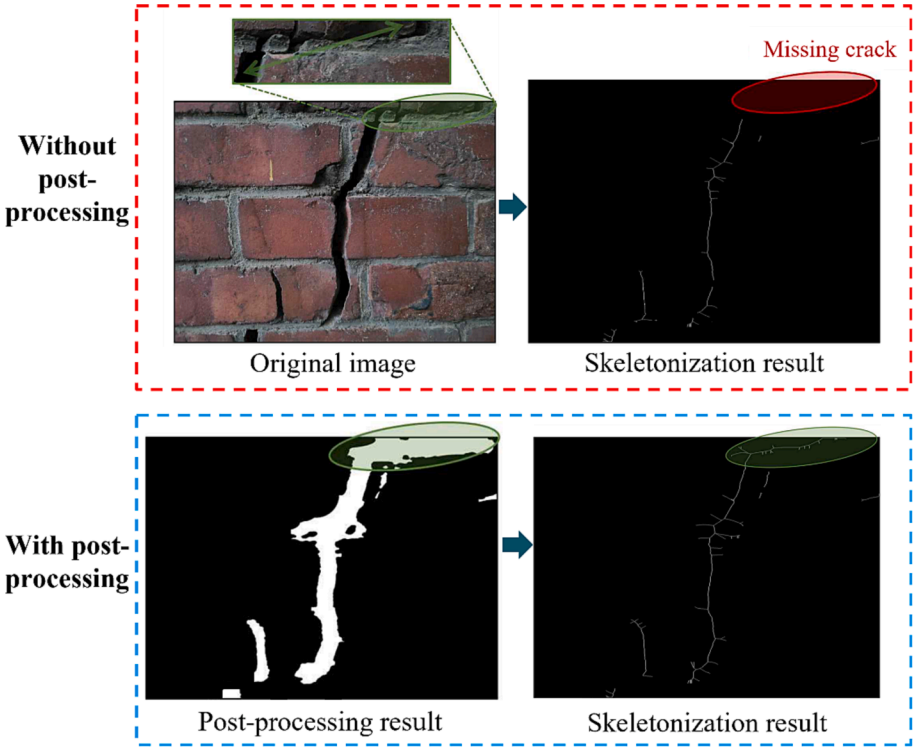

Fig. 11. The importance of the post-processing process, which reduces noises and missing crack problems.

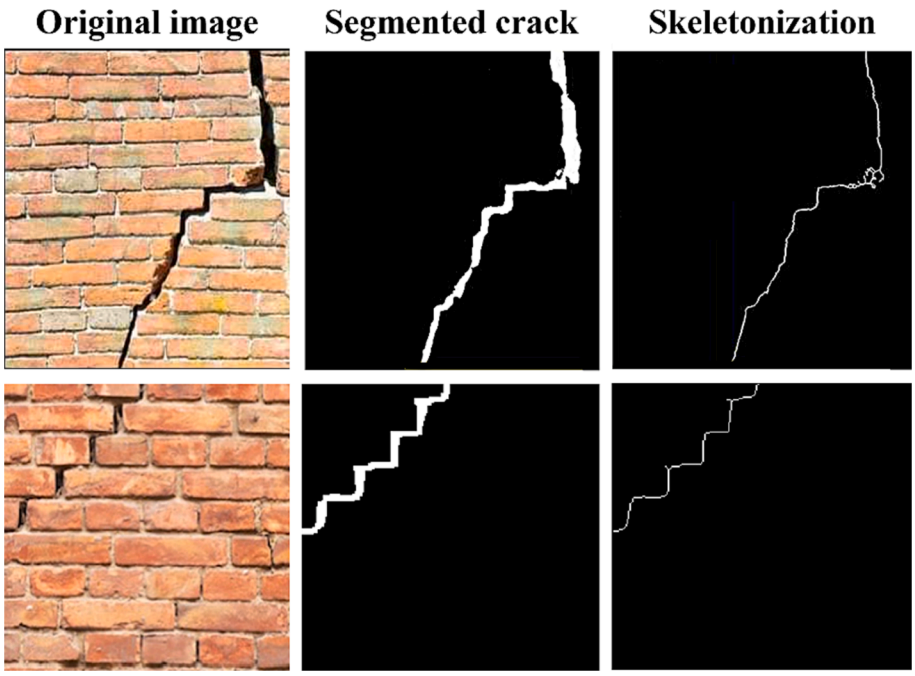

Fig. 12. Generated skeletons for two masonry crack samples.

corresponding skeletons for the segmented images, which is crucial for the crack length measurement technique to work correctly.

4.6. Real-life crack measurement

After extracting the crack skeleton from the segmented crack outputs, the crack length can be computed by the equation in Section 3.3. The computed crack's length is in pixels, which is challenging for the inspectors to decide its severity. As a result, brick detection and real measurement were performed to extract the real measurement in mm per pixel.

Fig. 13 describes the crack length measurement process in two different cases, simple crack (Fig. 13(a)) and complex crack (Fig. 13(b)). For each case, the original image, crack length measured by the ruler, predicted mask, and skeletonization result is displayed.

Based on the skeletonization output and the actual mm/pixel computed in Section 4.5, Table 4 shows a comparison between the actual inspection length and the model predicted length for the two particular cases.

In the first case (a) containing a single crack, the actual length of the crack is 260 mm. After the crack mask was predicted and the skeleton was extracted, the total number of computed crack length pixels was 1787, with the computed mm/pixel at 0.16. Therefore, the predicted crack length by the proposed framework is 285.9 mm, which leads to a measurement accuracy of 90.1 %.

For a more complex case (b) involving two separate cracks, the real combined crack length is 540 mm. The lengths of the first crack (#1) and the second crack (#2) predicted by the proposed framework were 485.4 mm and 112.5 mm, respectively. The total predicted crack length was 597.9 mm, so the accuracy for the second case is 90.3 %.

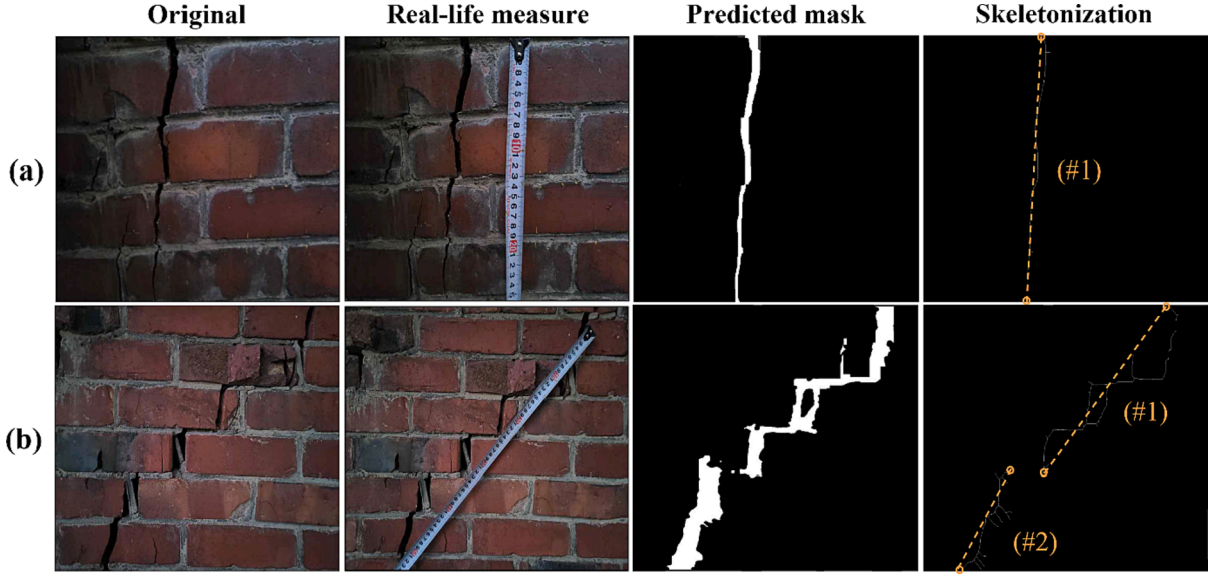

Fig. 13. Real crack length measurement for two different scenarios, (a) simple and (b) complex cracks.

Table 4
Comparison between the actual length inspection and the real length predicted by the framework.

Actual inspection			Model prediction				
Case	Real length (mm)	Crack	Predicted length (mm)	Pixels (px)	Computed mm/pixel	Error	Accuracy (%)
(a)	260	#1	285.9	1787	0.16	+25.9	90.1
(b)	540	#1	485.4	837	0.56	−57.9	90.3
		#2	112.5	194			
		Total	597.9	1131			

5. Discussion

Even though a limited number of studies have been introduced to identify masonry cracks, they were mostly classification [6,19] or detection [5,12]. In addition, the real-life crack measurement, a critical step for masonry inspection, was ignored. This research proposes a novel deep learning-based crack segmentation and measurement system that is distinguishable from previous approaches. The most crucial contribution is that the proposed framework can automatically identify masonry cracks' exact precise location/shape and provide real-life length measurements for the segmented cracks. The results of the methodologies and experiments are in good agreement qualitatively and quantitatively, indicating the results' reliability.

Various state-of-the-art segmentation models, such as DeepLabV3+, U-Net, and FPN, were trained on the proposed masonry segmentation dataset using different combinations of backbone structures and loss functions. The DeepLabV3 + model (ResNet backbone and BCE loss) showed the highest IoU score of 0.97, showing that the model precisely segmented cracks from input masonry images. Moreover, the DeepLabV3 + outperformed previous state-of-the-art crack segmentation models, including CrackNet and DeepCrack, which demonstrate the superiority of the proposed model.

Previous masonry crack identification frameworks provided the inspectors with basic crack information, such as crack class and location. However, they failed to provide context information, such as crack length, for determining the crack severity. This study efficiently solved this problem by first implementing a median-axis skeletonization algorithm in order to convert the segmented cracks into single-pixel crack skeletons to enable the computation of the crack length in pixels. However, it is still difficult for humans to comprehend the length in pixels, and the real-life measurement of a crack (in mm) can be computed based on the masonry brick units' fixed dimensionality and ratio nature. Therefore, we introduced a novel approach to compute the real-life length of the segmented cracks by detecting a brick unit using the Mask-RCNN model and performing real-life crack length extraction based on the algorithm explained in Section 3.2.1. Section 4.6 shows the effectiveness of the proposed approach for simple and complicated scenarios, which obtained real-life length high prediction accuracy of 90.1 % for simple and 90.3 % for complex crack scenarios.

6. Conclusion

This study presents an effective masonry crack detection and measurement framework based on deep learning. Through various experiments, the framework showed that structural cracks and brick unit detection could be performed with high performance. The framework significantly reduces the time and effort previously required for the inspectors to perform the inspection and documentation of the masonry structures while minimizing human errors at the same time.

The crack segmentation model based on DeepLabV3 + achieved an IoU score of 0.97 and an F1 score of 98 %. The quality of the output mask was then improved through various image-processing methods. After that, the medial-axis skeletonization method was then performed to obtain the crack skeleton that could be used for the crack measurement process. Notably, the proposed framework has a capacity not addressed in previous work; it provides real-life crack measurements through detecting and measuring the brick unit. The Mask-RCNN-based brick detection model showed a validation bounding box loss and mask loss at 0.16 and 0.1. The experimental results demonstrated that the model robustly detected stretcher bricks, header bricks, and bricks obscured by other objects. The detected bricks were then fed into a brick measurement algorithm in order to provide the real-life crack measurement in mm.

However, certain limitations remain in the proposed framework. Firstly, the dataset used in this framework mostly contains normal images and slightly skewed images (less than 20 degrees), and the real-life measurement extraction based on the brick unit part became less accurate when fed a highly skewed input image, leading to the wrong crack length measurement. Secondly, as the deep learning model automatically learns abstract features from the training dataset, the model will not detect crack features that do not appear in the dataset (complicated cracks, tiny cracks, etc.). As a result, based on the application use case, engineers must provide a sufficient number of images that contain the desired crack features.

CRediT authorship contribution statement

L. Minh Dang: Conceptualization, Methodology, Writing – review & editing. **Hanxiang Wang:** Methodology, Writing – review & editing. **Yanfen Li:** Visualization, Investigation. **Le Quan Nguyen:** Data curation. **Tan N. Nguyen:** Visualization, Investigation. **Hyoung-Kyu Song:** Funding acquisition. **Hyeonjoon Moon:** Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2020R1A6A1A03038540) and National Research Foundation of Korea (NRF) grant funded by the Korean government, Ministry of Science and ICT (MSIT) (2021R1F1A1046339) and by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, Forestry and Fisheries(IPET) through Digital Breeding Transformation Technology Development Program, funded by Ministry of Agriculture, Food and Rural Affairs(MAFRA)(322063-03-1-SB010).

References

[1] C. Scuro, et al., Internet of Things (IoT) for masonry structural health monitoring (SHM): overview and examples of innovative systems, Constr. Build. Mater. 290 (2021), 123092.
[2] D. Loverdos, V. Sarhosis, Automatic image-based brick segmentation and crack detection of masonry walls using machine learning, Autom. Constr. 140 (2022), 104389.

[3] T.N. Nguyen, L.M. Dang, J. Lee, P.V. Nguyen, Load-carrying capacity of ultra-thin shells with and without CNTs reinforcement, Mathematics 10 (9) (2022) 1481.
[4] T.N. Nguyen, J. Lee, L. Dinh-Tien, L. Minh Dang, Deep learned one-iteration nonlinear solver for solid mechanics, Int. J. Numer. Meth. Eng. 123 (8) (2022) 1841–1860.
[5] K. Chaiyasarn, W. Khan, Damage detection and localization in masonry structure using faster region convolutional networks, GEOMATE J. 17 (59) (2019) 98–105.
[6] D. Dais, et al., Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning, Autom. Constr. 125 (2021), 103606.
[7] S. Talebi, S. Wu, M. Al-Adhami, M. Shelbourn, J. Serugga, The development of a digitally enhanced visual inspection framework for masonry bridges in the UK, Constr. Innov. 22 (3) (2022) 624–646.
[8] N. Cavalagli, M. Gioffrè, S. Grassi, V. Gusella, C. Pepi, G.M. Volpi, On the accuracy of UAV photogrammetric survey for the evaluation of historic masonry structural damages, Procedia Struct. Integrity 29 (2020) 165–174.
[9] L.M. Dang, et al., Automatic tunnel lining crack evaluation and measurement using deep learning, Tunn. Undergr. Space Technol. 124 (2022), 104472.
[10] H. Wang, et al., Pixel-level tunnel crack segmentation using a weakly supervised annotation approach, Comput. Ind. 133 (2021), 103545.
[11] L.M. Dang, et al., Sensor-based and vision-based human activity recognition: a comprehensive survey, Pattern Recogn. 108 (2020), 107561.
[12] E. Valero, et al., Automated defect detection and classification in ashlar masonry walls using machine learning, Autom. Constr. 106 (2019), 102846.
[13] L.M. Dang, et al., DefectTR: End-to-end defect detection for sewage networks using a transformer, Constr. Build. Mater. 325 (2022), 126584.
[14] D. Minh, et al., Explainable artificial intelligence: a comprehensive review, Artif. Intell. Rev. (2021) 1–66.
[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. in: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
[16] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation. International Conference on Medical Image Computing and Computer-assisted Intervention, Springer, 2015.
[17] L.-C. Chen, et al., Encoder-decoder with atrous separable convolution for semantic image segmentation. Proceedings of the European conference on computer vision (ECCV), 2018.
[18] M. Ferguson, R. Ak, Y.-T. Lee, K.H. Law, Detection and segmentation of manufacturing defects with convolutional neural networks and transfer learning, Smart Sustain. Manuf. Syst. 2 (1) (2018) 20180033.
[19] M.J. Hallee, R.K. Napolitano, W.F. Reinhart, B. Glisic, Crack detection in images of masonry using cnns, Sensors 21 (14) (2021) 4929.
[20] Y. Li, et al., A robust instance segmentation framework for underground sewer defect detection, Measurement 190 (2022), 110727.
[21] T.Y. Zhang, C.Y. Suen, A fast parallel algorithm for thinning digital patterns, Commun. ACM 27 (3) (1984) 236–239.
[22] T.C. Lee, R.L. Kashyap, and C.-N. Chu, Building skeleton models via 3-D medial surface axis thinning algorithms. CVGIP: Graphical Models and Image Processing, 1994. 56(6): p. 462-478.
[23] S. Changxian, M. Yulong, Morphological thinning based on image's edges. ICCT'98. 1998 International Conference on Communication Technology. Proceedings (IEEE Cat No. 98EX243), IEEE, 1998.
[24] R. Marie, O. Labbani-Igbida, E.M. Mouaddib, The delta medial axis: a fast and robust algorithm for filtered skeleton extraction, Pattern Recogn. 56 (2016) 26–39.
[25] He, K., et al. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. 2017.
[26] J. Deng, et al., Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009.
[27] D.-H. Kim, et al., Lateral Force Resisting System of Flat Plate Structure based on KBC 2008 Draft. Proceedings of the Korea Concrete Institute Conference, Korea Concrete Institute, 2008.
[28] S. Qiu, et al., Methodology for accurate AASHTO PP67-10–based cracking quantification using 1-mm 3D pavement images, J. Comput. Civil Eng. 31 (2) (2017) 04016056.
[29] X. Yang, et al., Automatic pixel-level crack detection and measurement using fully convolutional network, Comput.-Aided Civ. Infrastruct. Eng. 33 (12) (2018) 1090–1109.
[30] Lin, T.-Y., et al. Feature pyramid networks for object detection. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
[31] Q. Zou, et al., Deepcrack: Learning hierarchical convolutional features for crack detection, IEEE Trans. Image Process. 28 (3) (2018) 1498–1512.
[32] A. Zhang, et al., Automated pixel-level pavement crack detection on 3D asphalt surfaces with a recurrent neural network, Comput.-Aided Civ. Infrastruct. Eng. 34 (3) (2019) 213–229.