Journal of Building Engineering 75 (2023) 106976

Contents lists available at ScienceDirect

Journal of Building Engineering

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

Full length article

CDD-TR: Automated concrete defect investigation using an improved deformable transformers

Minh Dang [a,b], Hanxiang Wang [c], Tri-Hai Nguyen [d], Lilia Tightiz [e], Liem Dinh Tien [f], Tan N. Nguyen [g], Ngoc Phi Nguyen [h,*]

[a] *Institute of Research and Development, Duy Tan University, Da Nang 550000, Viet Nam*
[b] *Faculty of Information Technology, Duy Tan University, Da Nang 550000, Viet Nam*
[c] *Department of Computer Science and Engineering, Sejong University, Seoul, Republic of Korea*
[d] *Faculty of Computer Science, Ho Chi Minh City Open University, Ho Chi Minh City, 70000, Viet Nam*
[e] *School of Computing, Gachon University, 1342 Seongnamdaero, Seongnam-si, Gyeonggi-do, 13120, Republic of Korea*
[f] *Faculty of Fundamental Sciences, Van Lang University, Ho Chi Minh city, Viet Nam*
[g] *Department of Architectural Engineering, Sejong University, 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea*
[h] *Department of Aerospace Engineering, Sejong University, Seoul, Republic of Korea*

ARTICLE INFO

Dataset link: https://www.aihub.or.kr/

Keywords:
Transformers
Deep learning
Concrete defect
Defect detection
Detr

ABSTRACT

Public infrastructures, such as bridges, dams, and buildings, play a key role in urban development. Structural inspection by visually monitoring and inspecting the structures for defects has become increasingly vital to prevent structural deterioration. However, previously, the structural inspection was primarily carried out manually, which was time-consuming, error-prone, and tedious. Therefore, this study proposes an efficient concrete defect detection system based on a transformer model. Four primary contributions are (i) a novel defect detection framework motivated by the deformable transformers (Deformable DETR); (ii) the use of a big concrete defect dataset containing four common defect types; (iii) multiple modules are introduced to the original Deformable DETR model and help the model achieve better performance; and (iv) visualization of the model's deformable attention weights to show the model effectiveness in detecting and localizing defects. The framework outperforms previous state-of-the-art object detection networks and obtains the mean Average Precision (mAP) of 63.8%.

1. Introduction

Structural assessment is crucial for ensuring the serviceability and structural integrity of public infrastructure, which has gained more attention recently [1]. Without proper structural assessment, defects can lead to severe structural failure, which results in costly repair and rehabilitation, and even causing enormous loss of human life [2]. Technological improvements such as vision sensors and unmanned aerial vehicles (UAVs) have been increasingly used to perform structural inspection. These improvements have effectively addressed some of the drawbacks of manual defect monitoring [3]. Nevertheless, the massive amounts of collected data require careful investigation by inspectors, causing a time burden that automation is intended to reduce [4,5].

Over the last few decades, machine learning (ML) and computer vision (CV) have been increasingly used in various domains, including structural inspection, agriculture, and autonomous driving. For example, ML has been used for detecting defects in concrete

[*] Corresponding author.
E-mail addresses: danglienminh@duytan.edu.vn (M. Dang), hai.ntr@ou.edu.vn (T.-H. Nguyen), liem.dt@vlu.edu.vn (L. Dinh Tien), npnguyen@sejong.ac.kr (N.P. Nguyen).

https://doi.org/10.1016/j.jobe.2023.106976
Received 14 March 2023; Received in revised form 26 May 2023; Accepted 30 May 2023
Available online 5 June 2023

structures using hand-crafted features such as gray feature [6], cascade feature [7], and V-shaped feature [8]. However, traditional ML models struggled to maintain high accuracy when dealing with images that differed slightly from the training set due to the hand-crafted feature engineering process [1,9].

Deep convolutional neural networks (CNNs) have emerged as a superior alternative to conventional ML in recent years, due to their remarkable performance in various CV tasks such as classification [10], detection [11], and segmentation [12,13]. For concrete crack detection, standard deep learning-based detection networks [14] typically use a pretrained CNN backbone (ResNet, VGGNet, and InceptionV3 [15]) on a large-scale benchmark dataset (such as ImageNet [16]) to obtain semantically robust abstract and coarse features. However, analysis has shown that current deep learning-based detection models rely heavily on hand-drafted parameters, such as anchors and proposals, to determine their performance [17]. Additionally, post-processing steps are needed to eliminate near-duplicate predictions.

Recently, transformers based on attention mechanisms have consistently delivered impressive performance on various natural language processing (NLP) tasks [18]. Transformers have recently been introduced to CV with the appearance of an end-to-end DEtection TRansformer (DETR) for object detection. The DETR's encoder–decoder structure eliminates the need for hand-crafted and time-consuming components in existing one-stage and two-stage object detection networks, such as anchor design and non-maximum suppression [19]. For example, Wan et al. recently introduced a deep learning model called Bridge Detection Transformers (BR-DETR) based on DETR, which uses a copy-paste data augmentation method and Deformable Conv2D to increase the sample size and replace convolution, respectively. The experimental results showed that BR-DETR outperformed SSD and YOLOv4 with the highest mAP of 95.7% on the augmented Shandong bridge damage dataset [20]. In another study, Zhang et al. suggested a hierarchical deep learning framework based on DETR to detect building façade defect with the IoU of 93.6% on wall components dataset [21]. However, it has been proven that DETR struggles with small objects [19] and has a long convergence time since it learns from a fixed feature spatial resolution [22]. To address these issues, an improved version of the DETR, called Deformable DETR, was proposed. Deformable DETR incorporates deformable attention mechanisms to enable more efficient learning and achieve higher detection rates for small objects [22].

Even though many deep learning-based models, such as YOLO and SSD, have been proposed for concrete defect detection, their performances can be significantly affected by manual parameter settings and post-processing processes. Furthermore, previous models were trained using small datasets collected under controlled environments, resulting in poor performance in real-life scenarios. Therefore, this study proposes an end-to-end transformer-based framework for detecting concrete defects by modifying certain components of the original deformable DETR model and training it on a large-scale dataset comprising over 200,000 images captured by a drone, covering four primary types of concrete defects. In addition, a mean feature map is extracted and visualized using the deformable attention weights from the last layers of both the encoder and decoder to efficiently interpret the model's performance.

This manuscript is organized as follows: Section 2 introduces the concrete defect detection dataset that was used in this study. Section 3 describes the overall framework of the concrete defect detection method. Section 4 explains each component of the framework in detail. Section 5 presents and evaluates the experimental results of the proposed system. Section 6 discusses the main findings and implications of this study. Section 7 concludes the paper and suggests directions for future work.

2. Concrete defect dataset

Previous studies on concrete defects have been limited in scope and scale, using small datasets with fewer than four defect types. For example, some studies focused on crack and non-crack [23–25], crack and rebar exposure [26], or crack, and spalling [27]. Only one study considered four defect types: crack, spot, rebar exposure, and spalling [14], but the dataset was small. In contrast, this research uses a large-scale dataset of over 200,000 images with four main defect types, surpassing most previous works regarding diversity and quantity. The dataset was provided by the National Information Society Agency of Korea (NIA) for research purposes.[1] It was mainly collected by the Safety and Quality Engineering Limited company[2] and labeled by the Korea Concrete Institute.[3]

The concrete defect dataset was collected by a commercial UAV DJI drone on various public infrastructures across South Korea in 2022. The UAV's max flight time is about 28 min. It has a high-resolution camera that supports up/down tilt. Each image has dimensions of 2560 by 1440 pixels at 72 dots per inch (dpi). The sample images for each concrete defect are illustrated in Fig. 1.

A definition of each type of defect is provided as follows.

- Crack: a vertical or diagonal defect emerges on the concrete surface due to the settling of the concrete foundation during concrete curing.
- Spalling: a typical issue for weak concrete surfaces exposed to damage leading to part of the surface peeling, breaking, or chipping away.
- Efflorescence: the appearance of white salt deposits on the surface of the concrete, which is caused by vapor migrating through the concrete slab.
- Exposed rebar: The exposed rebar is usually caused by concrete deterioration or errors during construction, which can weaken the concrete foundations.

[1] https://aihub.or.kr/

[2] http://www.sqeng.co.kr/

[3] https://www.kci.or.kr/

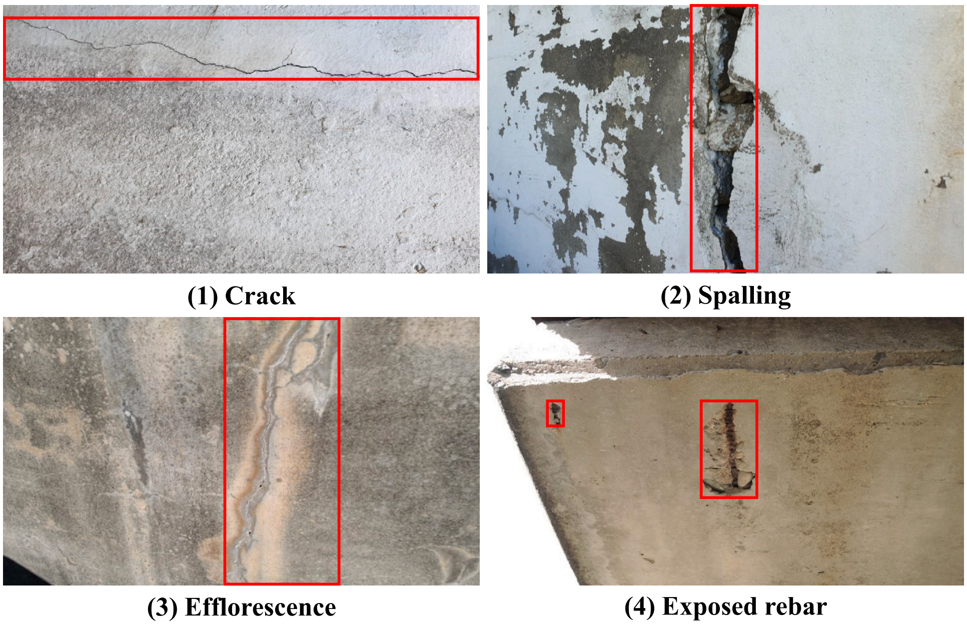


Fig. 1. Depiction of the four main types of concrete defect from the collected dataset.
Note: The red bounding box indicates the defect regions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

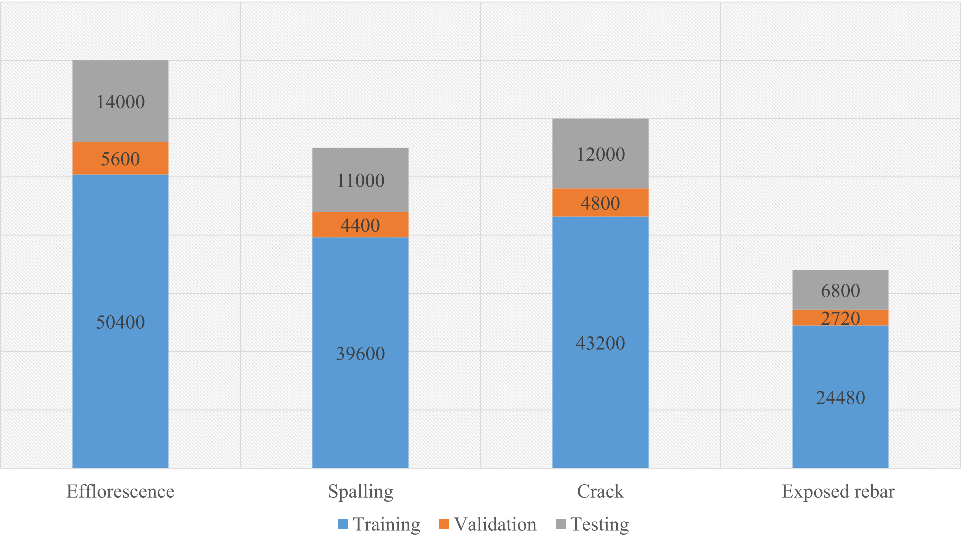


Fig. 2. A bar chart showing the number of images for each defect type.

Fig. 2 shows the total number of images annotated for each defect type. A total of 219,000 concrete defect images were collected. 175,200 images were randomly chosen as training data (80% of the original data). After that, 17,520 images (10% of the training data) were used as the validation data. Finally, 43,800 images were selected as testing data (20% of the original data).

3. System overview

Fig. 3 illustrates the primary components of an automated framework for concrete defect detection and analysis, which can be abbreviated as CDD-TR. Here, CDD represents concrete defect detection and analysis, and TR refers to deformable DETR.

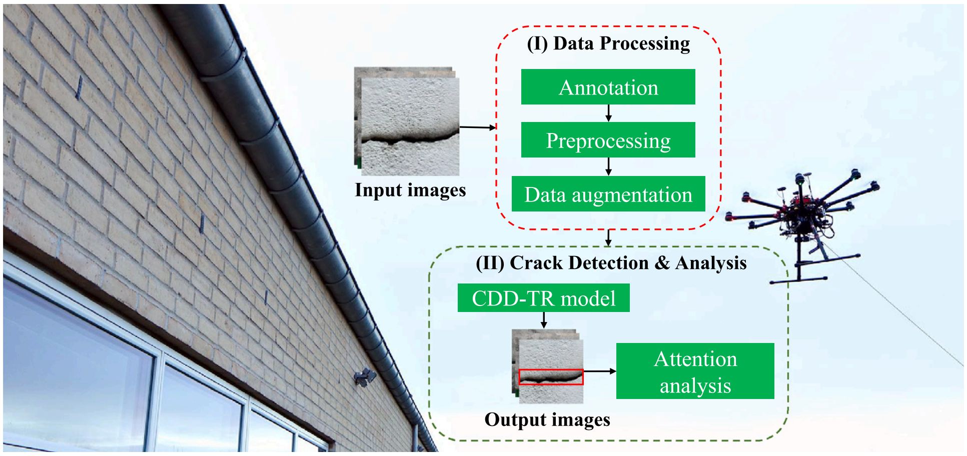


Fig. 3. Visualization for the main components of the proposed concrete defect detection framework (CDD-TR).

- Data processing: As the raw images collected by the drone may have quality issues, such as uneven brightness, darkness, or haziness, a pre-processing step is necessary to address these issues and remove any possible artifacts. This step enhances the image quality for better defect detection and analysis performance.
- Concrete defect detection: Despite achieving state-of-the-art performance on large-scale object detection datasets such as COCO Pascal VOC [28], current well-known object detection algorithms such as RCNN, YOLO, and SSD are complex, with multiple hyperparameters that require manual optimization, and have a more complicated training process than the end-to-end training style. This study modified a transformer-based object detection model, named deformable DETR [22], to efficiently identify concrete defects. It can be trained with an end-to-end approach.
- Attention analysis: Analyzing the attention weights of the transformer model used in the concrete defect detection framework can provide insights into the model's robustness in detecting defects. This information is valuable and can increase trust in the model's predictions. However, it is impossible to estimate the model's attention weights based on the bounding box information alone. Therefore, this research proposes to visualize and analyze the attention weights of the model to interpret its performance and gain insights into its behavior.

4. Methodology

4.1. Image pre-processing

Previous research has proved that various types of noise from the collected raw images can seriously impact the performance of the defect detection algorithm during the training phase [13,29]. Therefore, it is essential to implement some image pre-processing processes to improve image quality and defect detection efficiency.

Initially, a novel adaptive gamma correction with weighted histogram distribution (AGCWHD) suggested by [30] was applied to enhance the color preservation and contrast of raw images. AGCWHD was applied by first performing a contrast stretch for the input RGB images. Next, the pre-processed RGB images were converted into the hue, saturation, and intensity (HSI) color space. Finally, the proposed AGCWHD was implemented on the converted intensity channel to enhance the image contrast. AGCWHD demonstrated superior performance compared to previous contrast enhancement approaches in terms of colorfulness, entropy, and histogram utilization efficiency, without compromising image details.

The output images were then processed using a pretrained autoencoder and contrastive regularization network (AECR-Net) [31] to properly reduce noise before being fed into the model. The AECR-Net utilizes a novel contrastive regularization method to minimize the representation space distance between the estimated haze-free images and the ground truth (GT) images, while maximizing the distance between the estimated haze-free images and the input hazy images. The autoencoder network is enhanced with a dynamic feature enhancement approach and adaptive mixup operation to enable adaptive information flow preservation while expanding the receptive field to enhance the model's transformation power. AECR-Net demonstrated satisfactory denoising performance while preserving the input's initial brightness without requiring prior knowledge.

Fig. 4 shows four input images with various issues, such as blurriness, low-light conditions, and poor illumination, from the dataset, along with their corresponding outputs from the pre-processing module. After passing through AGCWHD and AECR-Net, the quality of the pre-processed images was significantly enhanced compared to their raw inputs. For example, cracks in the raw images with low-light or poor illumination conditions are difficult to observe, but the pre-processing module significantly improves the image quality, enabling better observation of the defects. Importantly, the pre-processing modules did not decrease the quality or add noise to the input images without any of the mentioned issues.

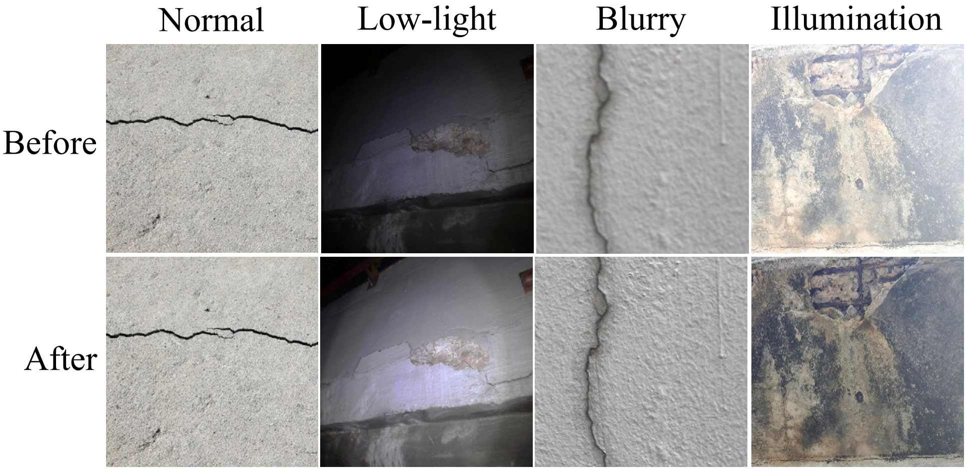


Fig. 4. Comparison of the four input images with and without the implementation of two pre-processing models, including contrast enhancement (AGCWHD) and denoising (AECR-Net).

4.2. Transformer-based defect identification

4.2.1. Attention mechanism

Self-attention forms the core module that enables the transformer to discover rich word-to-word relations and variations for each word [18]. This mechanism appears in both the encoder and decoder of the transformer models. In the encoder, this technique enables the input sequence to focus on itself, while in the decoder, it allows the target sequence to concentrate on itself. The self-attention is calculated using three distinctive vectors: query (Q), key (K), and value (V). These vectors are randomly initialized and multiplied by the embedding. The entire computation process can be described as follows.

$$\mathrm{Attention}(Q,K,V)=Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where d_k represents the dimensionality of the key vectors. To prevent the Softmax function from producing large values and to stabilize gradients, the score QK^T is divided by $\sqrt{d_k}$.

Migrating self-attention from sequence features to image feature maps presents a significant challenge due to the requirement of processing all possible spatial locations in the feature maps. Consequently, the algorithmic complexity increases as the size of the feature maps grows. To tackle this issue, Zhu et al. introduced a novel deformable attention mechanism that selectively attends to a few sampling points, independent of the spatial size of the input feature maps [22].

- Deformable attention: Let $x\in\mathbb{R}^{C\times H\times W}$ be an input feature map, where C is the number of channels, H is the height, and W is the width. Consider a query element denoted by q, with content feature $z_q\in\mathbb{R}^D$ and reference point index $p_q\in\mathbb{R}^2$. As stated in [22], the deformable attention feature can be represented as follows:

$$\mathrm{DeformAttn}\left(z_q,p_q,x\right)=\sum_{m=1}^{M}W_m\left[\sum_{k=1}^{K}A_{mqk}\cdot W_m' x\left(p_q+\Delta p_{mqk}\right)\right] \tag{2}$$

  In this equation, m is the attention head, k is the sampled keys, and K is the total number of sampled keys. The sampling offset is denoted by Δp_{mqk}, and the attention weight of the kth sampling point in the mth attention head is denoted by A_{mqk}. This can be computed using the query feature z_q, which is fed into a linear projection operator with three channels. Two sets of channels encode the sampling offsets, while one set encodes the attention weights.
- Multi-scale deformable attention: Previous studies have shown that the performance of object detection algorithms can be significantly improved using multi-scale features. Therefore, the deformable attention mechanism was extended to process multi-scale feature maps, which involve sampling LK points from the multi-scale feature maps instead of K points from single-scale feature maps. Drawing on the formulation presented in [22], this extension can be expressed as follows.

$$\mathrm{MSDeformAttn}\left(z_q,\hat{p}_q,\left\{u^l\right\}_{l=1}^{L}\right)=\sum_{m=1}^{M}W_m\left[\sum_{l=1}^{L}\sum_{k=1}^{K}A_{mlqk}\cdot W_m' u^l\left(\beta_l\left(\hat{p}_q\right)+\Delta p_{mlqk}\right)\right] \tag{3}$$

  where M represents the number of attention heads, while L denotes the number of multi-scale input feature levels. The normalized coordinates of the reference point for each query element q are denoted as $\hat{p}_q\in[0,1]^2$. Additionally, the function $\beta_l(\hat{p}_q)$ is responsible for re-scaling the normalized coordinates q to the input feature map of the lth level

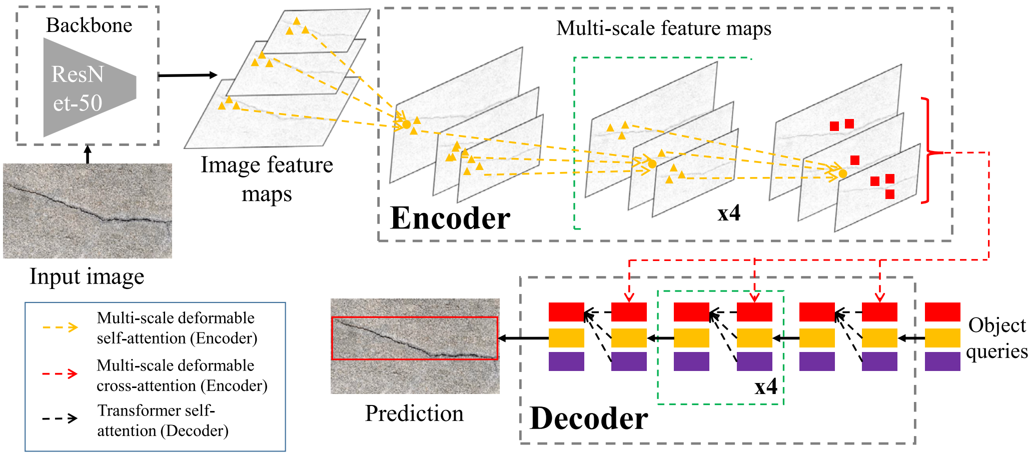

Backbone
ResN
et-50
Image feature maps
Multi-scale feature maps
Encoder
x4
Input image
Multi-scale deformable self-attention (Encoder)
Multi-scale deformable cross-attention (Encoder)
Transformer self-attention (Decoder)
Prediction
Decoder
x4
Object queries

Fig. 5. Description of the CDD-TR model, which is inspired by the deformable DETR model [22].

Table 1
Network details of the CDD-TR model.

Layer	Output size
Input	$H \times W \times 3$
Backbone (ResNet-50)	$H/32 \times W/32 \times 2048$
Deformable attention	$H/32 \times W/32 \times 2048$
Position embeddings	$H/32 \times W/32 \times d$
Transformer encoder	$Nq \times d$
Transformer decoder	$N \times (C + 4)$
Output	$N \times (C + 4)$

4.2.2. Deformable DETR

With a default level of $L = 4$, the multi-scale feature $\{x^l\}_{l=1}^{L}$ is obtained from the feature maps of C_3 to C_5 in the ResNet-50 backbone. Here, C_l indicates that the resolution of the extracted feature map is 2^l times smaller than the input image. These multi-scale feature maps are then fed into the encoder–decoder structure of deformable DETR, as illustrated in Fig. 5.

- Deformable transformer encoder: The key difference between deformable DETR and DETR is that deformable DETR replaces DETR's attention module with a multi-scale deformable attention module. Multi-scale feature maps are used as input and output to the encoder, and the key and query vectors utilize pixels from these feature maps. In deformable DETR, each query pixel's reference point is itself. The model also introduces positional embeddings with fixed encoding and scale-level embeddings (e_l) for feature representation. $\{e_l\}_{l=1}^{L}$ represents the feature level of each query pixel. These two techniques are randomly initialized and jointly trained with the network.
- Deformable transformer decoder: The decoder of the original DETR model consists of two main attention modules: self-attention and cross-attention. While the self-attention module remains unchanged, the cross-attention module is replaced by a multi-scale deformable attention module. This module focuses on extracting features around reference points, with key elements being extracted from the encoder's output multi-level feature maps. Bounding boxes are predicted based on offsets with reference points, which represent the center of the initially predicted bounding box.

Table 1 provides the network details of the CDD-TR model, including the output size for each layer.

The backbone network produces feature maps with a spatial resolution of $H/32 \times W/32$ and 2048 channels. Deformable attention is applied to these feature maps to better handle object deformations. Position embeddings are added to the feature maps before passing them to the transformer encoder, which generates a set of object queries with a size of $Nq \times d$, where Nq is the number of object queries and d is the dimensionality of the queries. The transformer decoder takes in the feature maps, object queries, and position embeddings, and generates a set of object predictions with a size of $N \times (C+4)$, where N is the number of predicted objects, C is the number of object classes, and 4 represents the four coordinates of the bounding box. Finally, the output of the model is a set of object predictions with a size of $N \times (C + 4)$.

Although deformable DETR can be directly applied to detect concrete defects, certain components of the model may affect its performance. In order to enhance the performance of deformable DETR in concrete defect detection, several improvements to the model architecture and optimization process were introduced to the CDD-TR model and are described in Table 2. It is worth noticing that the modifications were carefully selected based on their effectiveness in enhancing the performance of transformer-based models in CV tasks [32,33].

Table 2
Changes introduced to the CDD-TR model in order to improve the performance of the original deformable DETR.

Model	Augmentation	Activation	Loss	Optimizer
Deformable DETR	\\	ReLU	GIoU	Adam
CDD-TR	Colorjitter	LeakyReLU	CIoU	LaProp [33]

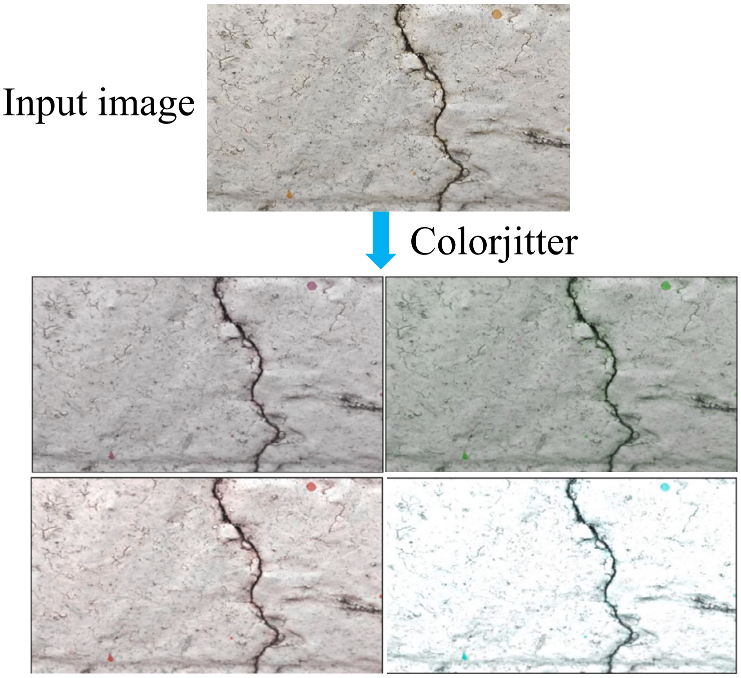


Fig. 6. Four output samples for an input image using the Colorjitter method.

By leveraging the strengths of Deformable DETR and incorporating these targeted modifications, the CDD-TR model demonstrated significant improvements in detecting concrete defects. The four main modifications listed in Table 2 can be explained in detail as follows:

- Colorjitter: Colorjitter augmentation technique has been utilized to create output images by randomly adjusting the brightness, contrast, saturation, and hue of an input image. The range factors for brightness, contrast, and saturation are required to be non-negative, while the range factor for hue must be less than 0.5. For this study, the range factors were set to [0, 2] for contrast, [0.5, 1.5] for brightness, [0.9, 1.1] for saturation, and [−0.5, 0.5] for hue. Fig. 6 illustrates four random outputs of the Colorjitter augmentation.
- Leaky Rectified Linear Unit (LeakyReLU): The original ReLU activation used in deformable DETR encountered the “dying” ReLU problem, where neurons became inactive for certain inputs and hinder gradient flow during training. To address this issue, this study replaced the ReLU activation with LeakyReLU activation [34], which incorporates a small negative slope to alleviate the “dying” ReLU problem and enhance training optimization. LeakyReLU introduces non-linearity and diversity to the model output, thereby augmenting its expressive power.
- Complete Intersection over Union (CIoU) loss: The Generalized IoU (GIoU) loss [35] employed in deformable DETR gradually expands the predicted box over multiple iterations to align with the GT box. In this study, the GIoU loss was replaced with the CIoU loss, which leverages three geometric parameters (central point, overlap area, and aspect ratio) between the predicted box and GT box to precisely align them. The CIoU loss exhibits faster convergence and achieves better detection performance compared to the GIoU loss [36].
- LaProp optimizer: While Adam is commonly selected as the default optimizer for many deep learning-based models, Ziyin et al. highlighted potential issues of instability and divergence during training caused by a mismatch between adaptivity parameters and momentum [33]. Therefore, in this study, the LaProp optimizer is used as an alternative. LaProp combines the Adam and Lookahead optimizers by utilizing Lookahead to update Adam’s momentum term. It has been demonstrated to offer improved stability and faster convergence compared to Adam across different benchmark datasets.

4.3. CDD-TR's attention analysis

Once the training phase was completed, the attention weights learned by the final layers of the encoder and decoder (averaged across all multi-scale prediction heads) were extracted. These weights were used to visualize the areas of an object that the transformer model focused on, which had high attention weight, in order to predict a specific class. The attention weights were stored as a square matrix with dimensions of $[H \times W, H \times W]$. To facilitate the illustration of attention feature maps, the matrix was converted to $[H, W, H, W]$. Subsequently, a mean attention feature map was computed using both the encoder and decoder attention feature maps. This map can be used to evaluate the model's performance in identifying defects and assessing its robustness.

5. Experimental results

This section presents a series of experiments conducted on the collected concrete defect dataset to thoroughly evaluate the performance of the CDD-TR model in various testing scenarios. Section 5.1 outlines the evaluation metrics used to assess the model's performance in different aspects, while Section 5.2 describes the hardware and environment in which the model was implemented. Additionally, it provides an explanation of the hyperparameters used for various models.

Section 5.3 comprises a group of experiments that were conducted to thoroughly evaluate the proposed model. In Section 5.3.1, the improvement in defect detection performance due to the pre-processing stage is assessed. Section 5.3.2 examines the influence of various modules proposed in the deformable DETR model. Section 5.3.3 explains how deformable attention extraction works and demonstrates that the model correctly focuses on relevant defect regions [37]. In Section 5.3.4, the CDD-TR model was qualitatively evaluated on four defect types, and challenging scenarios were also described. Finally, Section 5.3.5 compares the performance of CDD-TR with other models.

5.1. Evaluation metrics

Three components of the confusion matrix – true positive (TP), false negative (FN), and false positive (FP) – are used to evaluate the effectiveness of the concrete defect identification framework. These measures are utilized to calculate precision, recall, and mean average precision (mAP). The mAP metric is employed as the primary evaluation criterion as it considers all defect classes in the proposed dataset. In this study, the mAP metric uses an Intersection over Union (IoU) threshold of 0.5 to assess the accuracy of object detection. This means that the model considers a detection to be correct if the IoU between the predicted bounding box and the GT bounding box is greater than or equal to 0.5.

$$mAP = \frac{1}{N_{\text{classes}}} \sum_i AP_i \tag{4}$$

where i represents a defect class out of N_{classes}, which is four in this study.

The importance of precision and recall as critical metrics for practical concrete defect detection cannot be overstated. Precision measures the rate of incorrect detections, while recall measures the rate of missed detections. Therefore, in addition to computing mAP metrics, precision and recall are also calculated, and they can be expressed as follows:

$$Precision = \frac{TP}{TP + FP}; Recall = \frac{TP}{TP + FN} \tag{5}$$

5.2. Implementation descriptions

The concrete defect identification system was developed and trained using PyTorch,[4] a Python ML library. To ensure reliable experiments, a pre-trained ResNet-50 model on ImageNet [16] was used as the backbone for all detection models. The training and testing processes were deployed on two Nvidia Tesla V100 GPUs, each with 32 GB of memory. Except for the CDD-TR model, other deep learning models and their hyperparameters were implemented using open-source code provided by the original papers.

The encoding and decoding layers in both the deformable DETR and CDD-TR models were kept at a fixed number of 6. Both models had 8 attention heads with 256 hidden feature dimensions. CDD-TR's encoder parameters were shared across various feature levels, and the number of query slots was set to 300. The hyperparameters and training strategies were similar to those of deformable DETR, except for the use of cIoU instead of GIoU for the bounding box regression loss with $\beta = 1$ and $\gamma = 0.2$. The CDD-TR model was trained for 2 million iterations with a LaProp optimizer, using an initial learning rate of $4e - 4$, $\mu = 0.9$ and $\nu = 0.999$, and weight decay of $e{-}4$. A simple learning rate scheduler following the recommended schedule by [19,22], was adopted with a learning decayed at the 1.6 million iterations by a factor of 0.1.

5.3. Defect identification performance assessment

5.3.1. Pre-processing module analysis

The performance of seven object detection models (SSD [38], YOLOv5 [39], Faster R-CNN [40], Mask-RCNN [41], DETR [19], deformable DETR [22], and CDD-TR (ours)) was compared on the concrete defect testing set with and without implementing the pre-processing module. The four-class concrete defect dataset used in this study was labeled using COCO and can be easily adapted for training these models. In general, the seven models trained on pre-processed images achieved higher detection metrics than those trained without the pre-processing module, as shown in Table 3.

[4] https://pytorch.org/

Table 3
The impact of the pre-processing module on various models' performance.

Approach	Model	mAP (%)	Precision (%)	Recall (%)
Raw input	YOLOv5 [39]	53.1	55.9	56.1
	SSD [38]	48.3	51.7	53.5
	Faster R-CNN [40]	53.6	54.4	58.4
	Mask-RCNN [41]	51.2	51.8	52.6
	DETR [19]	53.5	55.3	56.1
	Deformable DETR [22]	55.7	56.5	55.1
	CDD-TR (Ours)	**62.1**	**58.2**	**57.9**
Pre-processed input	YOLOv5 [39]	55.3	55.7	56.1
	SSD [38]	49.8	50.2	54.1
	Faster R-CNN [40]	54.1	53.7	58.9
	Mask-RCNN [41]	53.5	53.6	54.1
	DETR [19]	55.1	55.9	57.2
	Deformable DETR [22]	58.5	56.9	57.4
	CDD-TR (Ours)	**63.8**	**60.4**	**58.9**

Table 4
Ablation study of the CDD-TR model.

	ColorJitter	Laprop	LeakyReLU	CIoU	mAP
CDD-TR (i)	✓				60.1
CDD-TR (ii)	✓	✓			60.9
CDD-TR (iii)	✓	✓	✓		63.3
CDD-TR (iv)	✓	✓	✓	✓	**63.8**

The pre-processing module resulted in a considerable increase in mAP of 2%–3% for most models, with the exception of deformable DETR and the proposed CDD-TR, which exhibited a slight mAP improvement of less than 1%. The efficiency and robustness of the proposed CDD-TR model were proven with the highest mAP and recall of 63.8% and 58.9%, respectively. Moreover, the proposed CDD-TR model outperformed other detection models and the original deformable DETR model by a margin of 4.3% mAP. It can be concluded that the pre-processing process is essential, particularly for outdoor videos, in enhancing the performance of concrete defect detection.

Fig. 7 shows that both versions of the CDD-TR model, represented by orange for raw images and blue for pre-processed images, displayed a swift drop in training and validation class error to approximately 25 after 200,000 iterations. The class error values then slowly declined and stabilized at less than 15 after 2 million iterations. The pre-processed version exhibited slightly better training and validation class error values than the raw version. Both models' mAP values rapidly reached 50% after 200,000 iterations, gradually increased and leveled off at 63.8% for the pre-processed version and 62.1% for the raw version. The recall metric followed a similar trend as the mAP value. Notably, at 1.6 million iterations, there is a sharp rise in mAP and a decrease in class error, primarily due to a scheduled change in the learning rate to a smaller value, resulting in more stable and accurate model training.

The preprocessing process showed that it improved the performance of concrete defect detection compared to using raw images. Therefore, all subsequent experiments were conducted using the preprocessed images.

5.3.2. Ablation study

An ablation study was conducted in this section to examine how different modules affect the deformable DETR model on the concrete defect dataset. The results are reported in Table 4. CDD-TR (i) differs from the original deformable DETR model only by adding a ColorJitter augmentation method, which improves the mAP by 1.6% to 59.1% compared to the original model. CDD-TR (ii) uses the Laprop optimizer instead of the Adam optimizer and converges about 20% faster (400k iterations less), resulting in a shorter training process. CDD-TR (iii) adds LeakyReLU activation to CDD-TR (ii) and boosts the mAP by 3.2% to 63.3% compared to the baseline model CDD-TR (i). Finally, CDD-TR (iv) replaces GIoU with CIoU for the bounding loss and achieves the highest mAP of 63.8%. Therefore, by modifying some network components, the CDD-TR model outperforms the original deformable DETR model by 3.7% in mAP.

5.3.3. Multi-scale deformable attention

To demonstrate the effectiveness of the CDD-TR model in detecting concrete defects, multi-scale deformable attention-weight feature maps were extracted from both the encoder and decoder of the CDD-TR structure. A mean attention-weight feature map was then computed using these extracted feature maps and overlaid on the RGB image for comparison. As shown in Fig. 8, the CDD-TR model, which utilizes the attention mechanism, accurately focuses on the defect areas, demonstrating its efficacy in concrete defect detection.

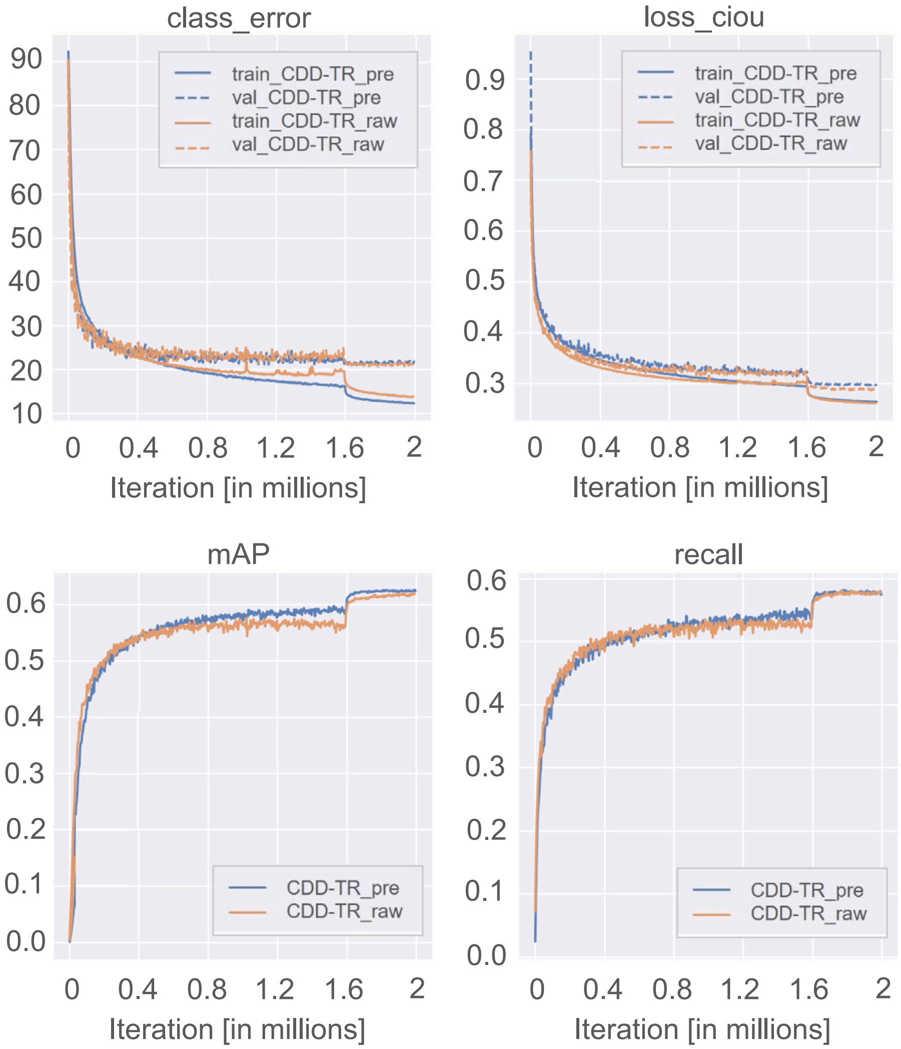


Fig. 7. Class error, CIoU loss, mAP, and recall logs of two different version of the proposed CDD-TR model trained on raw and pre-processed datasets.

Table 5
Performance of the CDD-TR for each concrete defect type (mAP, precision, and recall).

	Efflorescence	Spalling	Crack	Exposed rebar
mAP	55.1	57.2	68.7	63.4
Precision	52.9	58.3	66.5	63
Recall	50.4	59.2	69.3	63.7

5.3.4. CDD-TR performance evaluations

Fig. 9 illustrates the outputs of the CDD-TR model for four types of concrete defects, including the bounding box and confidence. The model accurately detected and localized each defect, and the mean deformable attention weights demonstrated its focus on the relevant regions for class identification. Notably, Fig. 9(b) and (d) showcase the model's ability to distinguish between multiple instances of efflorescence and exposed rebar, highlighting the robustness of the proposed model.

Table 5 presents the performance of our model for each concrete defect type in terms of mAP, precision, and recall using the testing set. The results indicate that our model achieved the highest performance in crack detection, with an mAP of 68.7%, a precision of 66.5%, and a recall of 69.3%. Conversely, the model exhibited the lowest performance in efflorescence detection, with an mAP of 55.1%. The performance for the other two defect types, spalling and exposed rebar, was relatively similar, ranging from 57.2% to 63.7%.

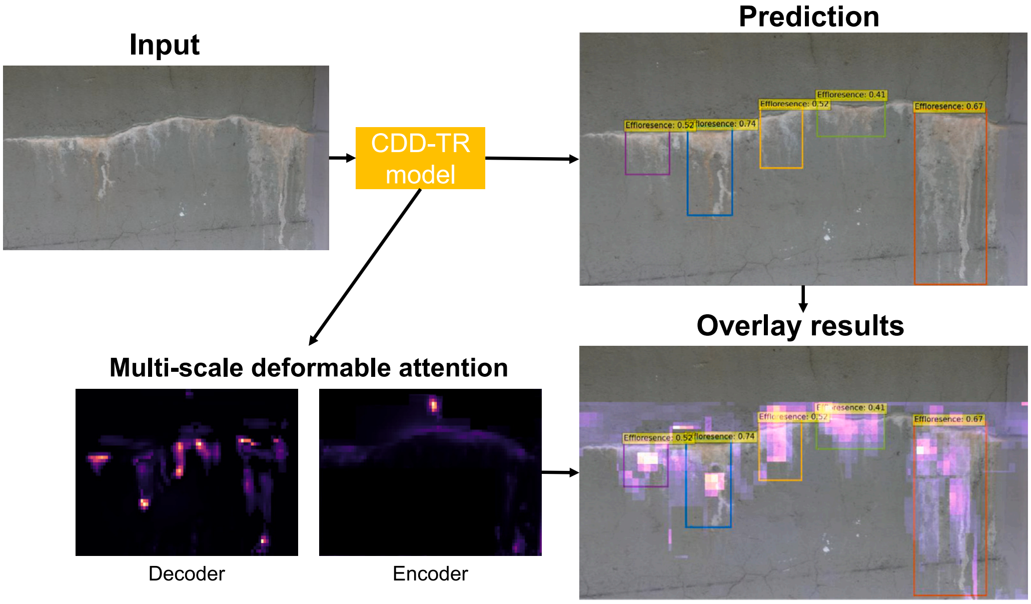
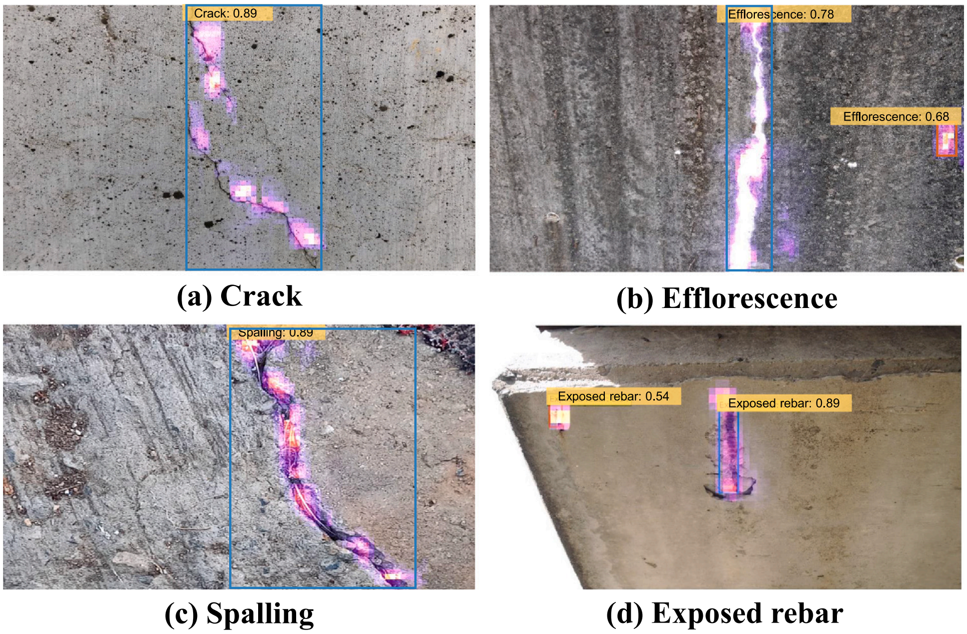


Fig. 8. Sample visualization of the deformable attention extraction process.
Note: The model output for each prediction contains the defect class and the confidence level of the model in that prediction.



(a) Crack

(b) Efflorescence

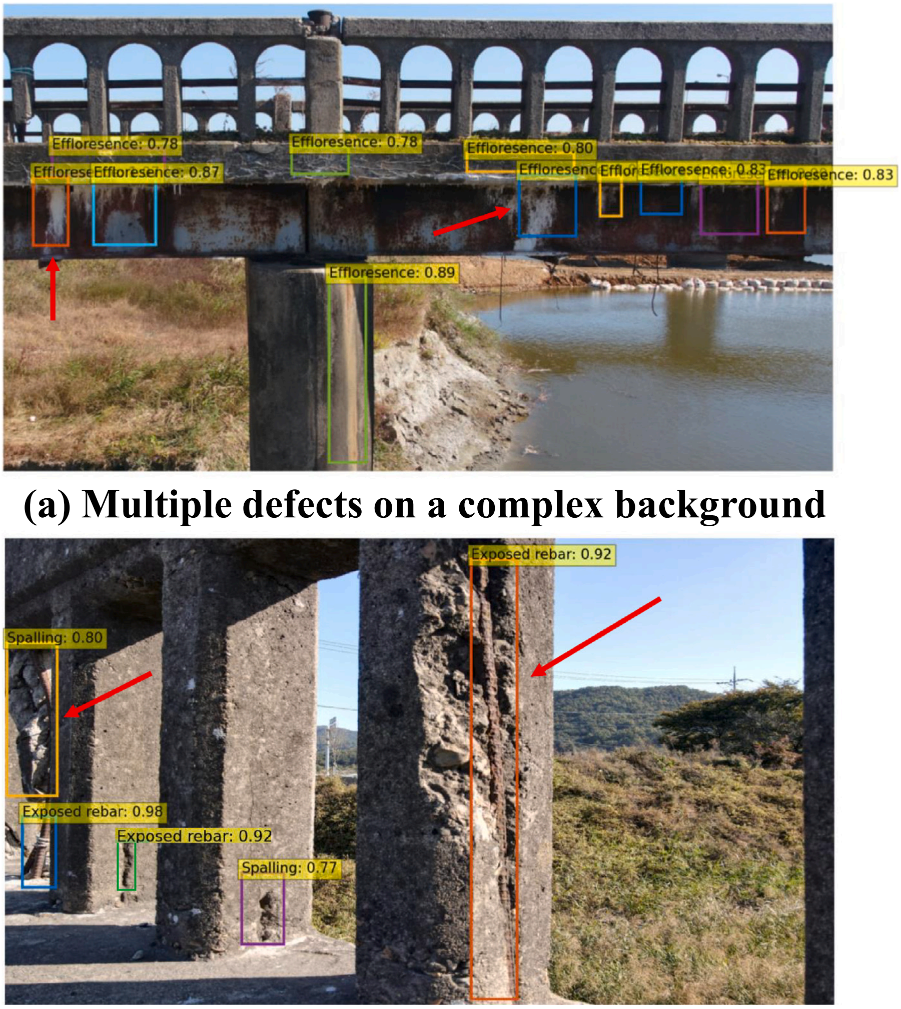
(c) Spalling

(d) Exposed rebar

Fig. 9. Predictions of the CDD-TR model for each defect type include the input image, defect location represented by the bounding box, and visualization of attention weights in the form of a mean feature map.
Note: The model output for each prediction contains the defect class and the confidence level of the model in that prediction.

One possible reason for the relatively poor performance in efflorescence detection is its limited contrast against the background, especially on concrete surfaces with a light color. Efflorescence could also exhibit variations in thickness and shape, making it challenging to distinguish from other surface features. Furthermore, it may be prone to confusion with accumulated dust or dirt on the concrete surface over time. Similarly, there could be misclassifications between spalling and exposed rebar since both involve the visibility of rebar on the surface.

Furthermore, the model achieved an average mAP of 61.1%, as presented in Table 5. The main contributing factor to this performance is the large and challenging nature of the dataset used in this study, which was captured by a drone under real-life



(a) Multiple defects on a complex background

(b) Multiple defect types at the same position

Fig. 10. Two image cases where the defects are difficult to identify due to their ambiguous appearance.
Note: The model output for each prediction contains the defect class and the confidence level of the model in that prediction.

conditions. Fig. 10 showcases two images that exemplify the ambiguous appearance of defects in concrete. In Fig. 10(a), the side of a concrete bridge exhibits various efflorescence spots. Although the model correctly detected the primary efflorescence on the bridge pier, other efflorescence defects were not labeled in the GT image, resulting in false alarms. Fig. 10(b) illustrates another scenario where some defects are distant from the camera, and the exposed rebar and spalling defects appear in the same location, potentially causing confusion for the model.

Fig. 11 illustrates the robustness of our model in handling challenging cases. In the first case (a), the model correctly detects an exposed rebar defect that is small and blends with the background in a low-light environment with a confidence of 0.92. For the second case (c), although the model correctly identifies an exposed rebar defect with high confidence, it wrongly detects the iron frame as exposed rebar due to its resemblance to the defect. In the third case (b), the model identifies multiple efflorescence and crack defects despite the blurriness and sunlight interference. The pre-processing module enhances the image quality and helps the model perform well. In the final case (d), the model recognizes multiple concrete cracks from a highly blurry image, again with the help of the pre-processing module. These results demonstrate the robustness of our proposed model in detecting concrete defects under various challenging conditions.

5.3.5. *Comparison study for CDD-TR*

The main goal of this section is to demonstrate the strength of the proposed CDD-TR model compared to other detection models, such as SSD [38], YOLOv5 [39], Faster R-CNN [40], Mask-RCNN [42], DETR [19], and the original deformable DETR [22]. Table 6

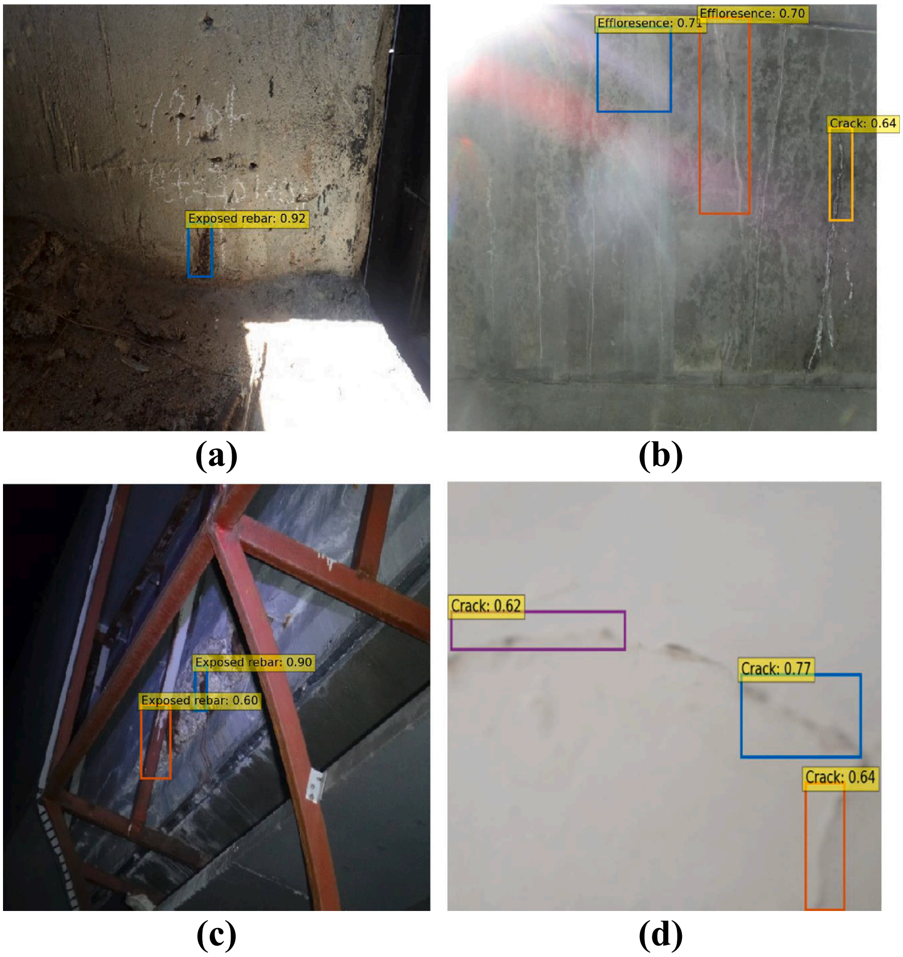


Fig. 11. Predictions of the proposed CDD-TR model for challenging scenarios.
Note: The model output for each prediction contains the defect class and the confidence level of the model in that prediction.

Table 6
Performance of CDD-TR model compared to the other approaches on the testing dataset.

Model	mAP	Precision	Recall	Speed (ms)
SSD [38]	48.3	45.1	46.9	**30**
YOLOv5 [39]	53.7	55.4	54.2	45
Faster-RCNN [40]	52.5	51.7	53.6	106
Mask-RCNN [41]	50.1	49.8	51.2	2800
DETR [19]	57.8	57.5	58.4	150
BR-DETR [20]	58.7	59.4	60.1	150
Deformable DETR [22]	58.1	59.3	57.8	200
CDD-TR (Ours)	**61.1**	**60.2**	**60.7**	175

shows their performances in terms of precision, recall, mAP, and inference time per image. Higher values of mAP, precision, and recall indicate more accurate and complete detection, while lower values of inference time suggest faster processing speed.

Among the models in the table, CDD-TR achieves the highest mAP (61.1%), precision (60.2%), and recall (60.7%), while also having a relatively fast speed (175 ms). This indicates that CDD-TR is an effective and efficient model for object detection. CDD-TR is based on deformable DETR, which also performs well on mAP (58.1%), precision (59.3%), and recall (57.8%), but has a slightly slower speed (200 ms). In comparison, the original DETR show the highest mAP of 57.8% with a detection speed of 150 ms. Finally, the BR-DETR model, which was proposed for concrete damage detection, shows a relatively better performance than the deformable DETR with the mAP at 58.7% and a faster detection speed of 150 ms.

The other models in the table are SSD, YOLOv5, Faster-RCNN and Mask-RCNN. SSD has the fastest speed (30 ms), but has a lower mAP at 48.3% than CDD-TR and deformable DETR. YOLOv5 has a similar speed to SSD (45 ms), but has a higher mAP at 53.7%. Faster-RCNN has a higher mAP of 52.5% than SSD and YOLOv5, but has a slower speed (106 ms). Mask-RCNN has a similar mAP at 50.1% to Faster-RCNN, but has the slowest speed (2800 ms). Mask-RCNN can also perform instance segmentation, which is an additional task that the other models cannot do. To sum up, the table demonstrates that CDD-TR outperforms all the other models in terms of mAP and speed.

6. Discussion

In this study, our aim was to identify the most effective framework for concrete defect detection. We evaluated seven different deep learning-based detection models on a large concrete defect dataset using three evaluation metrics: mAP, precision, and recall. Our key finding was that transformer-based models outperformed other standard detection models when trained on large concrete defect datasets. These models demonstrated the ability to capture more complex features and learn better representations of defect regions. Our findings align with previous research that has also demonstrated the effectiveness of transformer-based models, such as DETR [19] and deformable DETR [22]. However, it is important to note that these models have some limitations, including longer training times and lower inference speeds, which may make them less suitable for real-time detection scenarios. On the other hand, models like YOLOv5 and Faster R-CNN are better suited for applications that require fast detection, prioritizing speed over performance. Nevertheless, all models were able to generate predictions within a second after training, making them viable for concrete defect detection tasks.

To further enhance the performance of the transformer-based framework, we introduced the CDD-TR model for concrete defect detection by incorporating various modules into the original deformable DETR model. A key addition was the pre-processing module, which played a crucial role in handling datasets captured by drones in outdoor environments. Previous studies have highlighted the significant influence of noise on model performance, particularly in diagnosing cracked surfaces [13,29]. However, they did not provide a viable solution. In this study, we utilized the AGCWHD and AECR-Net models to preprocess the training images, resulting in a 1.7% increase in the mAP of the CDD-TR model. Other models also benefited from this module, exhibiting improved detection performance ranging from 2%–4%. While the pre-processing module requires additional computational power and time, it can be easily enabled or disabled based on the specific application's requirements. Overall, the proposed pre-processing module enables more effective detection of concrete defects in challenging outdoor environments.

In addition to the pre-processing module, we made several minor modifications to the original deformable DETR model (Section 4.2). These changes involved replacing ReLU with LeakyReLU activation, utilizing CIoU loss instead of GIoU loss, and adopting the LaProp optimizer instead of Adam optimizer. While these modifications were not specifically tailored for concrete defect detection, they are widely employed in the CV field due to their proven effectiveness [32,33]. Through the ablation study conducted in Section 4.3, we demonstrated that these changes improved the mAP value of CDD-TR by 3.7% to 63.8%. Notably, these modifications were straightforward to implement and resulted in faster convergence and enhanced generalization performance without compromising the detection capability.

Despite the numerous deep learning-based defect detection studies proposed in recent years, which have achieved satisfactory performance [24,43], they have often overlooked the crucial aspect of interpreting the models' predictions. In the field of civil engineering, this interpretability is essential for enhancing user confidence. This study places significant emphasis on the importance of model interpretability in automated concrete evaluation systems and highlights the interpretability of our proposed CDD-TR model, which is based on the transformer architecture. Unlike other models such as YOLOv5, SSD, or Faster-RCNN, the CDD-TR model enables the extraction and visualization of deformable attention weights from both its encoder and decoder (Section 5.3.3). This unique feature facilitates the understanding of the model's predictions and enhances transparency, which plays a vital role in fostering trust in automated concrete evaluation systems.

7. Conclusions and future works

This study introduces an end-to-end transformer-based model for concrete defect identification, which can be applied to concrete inspection applications. The model was trained on a dataset comprising 219,000 images of four types of concrete defects. Various modifications, including the utilization of the LaProp optimizer, the application of colorjitter augmentation, the implementation of the CIoU loss, and the adoption of LeakyReLU activation, were incorporated to enhance the performance of the original deformable DETR model. The results demonstrate that the proposed model achieves high accuracy and speed in detecting concrete defects. Furthermore, this paper reveals the usefulness of attention weights, a unique feature of the transformer model, in understanding how the model identifies defect regions.

The suggested framework robustly detects four types of concrete defects with a high mAP of 63.8%, surpassing six other standard object detection models based on the results of a series of experiments. The mAP value also exhibits an improvement from 60.1% to 63.8% compared to the original deformable DETR model, thanks to the inclusion of the pre-processing module and the modifications made to the loss function and optimizer. Additionally, the transformer attention weights offer valuable insights into the model's decision-making process by highlighting the relevant defect areas for accurate predictions.

While the concrete defect dataset used in this study contains four common defect types, the addition of more defect classes could further enhance the detection performance. Furthermore, it would be valuable to develop a concrete defect severity standard to guide the analysis of detected defects. As transformer-based models have a complex structure, their inability to detect defects in real-time is a limitation that needs to be addressed in future work. Thus, optimizing these models for both robustness and time efficiency is a crucial direction for future research.

CRediT authorship contribution statement

Minh Dang: Writing – original draft. **Hanxiang Wang:** Formal analysis, Conceptualization, Visualization. **Tri-Hai Nguyen:** Investigation, Data curation. **Lilia Tightiz:** Software, Writing – review & editing. **Liem Dinh Tien:** Data Curation, Validation. **Tan N. Nguyen:** Supervision. **Ngoc Phi Nguyen:** Funding acquisition.

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

The data is publicly available on https://www.aihub.or.kr/.

References

[1] D. Ai, G. Jiang, S.-K. Lam, P. He, C. Li, Computer vision framework for crack detection of civil infrastructure—A review, Eng. Appl. Artif. Intell. 117 (2023) 105478.

[2] L.M. Dang, H. Wang, Y. Li, L.Q. Nguyen, T.N. Nguyen, H.-K. Song, H. Moon, Deep learning-based masonry crack segmentation and real-life crack length measurement, Constr. Build. Mater. 359 (2022) 129438.

[3] L.M. Dang, H. Wang, Y. Li, Y. Park, C. Oh, T.N. Nguyen, H. Moon, Automatic tunnel lining crack evaluation and measurement using deep learning, Tunn. Undergr. Space Technol. 124 (2022) 104472.

[4] 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.

[5] T.N. Nguyen, J. Lee, L. Dinh-Tien, L. Minh Dang, Deep learned one-iteration nonlinear solver for solid mechanics, Internat. J. Numer. Methods Engrg. 123 (8) (2022) 1841–1860.

[6] S. Liang, X. Jianchun, Z. Xun, An extraction and classification algorithm for concrete cracks based on machine vision, IEEE Access 6 (2018) 45051–45061.

[7] R. Ali, D.L. Gopal, Y.-J. Cha, Vision-based concrete crack detection technique using cascade features, in: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018, Vol. 10598, SPIE, 2018, pp. 147–153.

[8] Y. Sato, Y. Bao, Y. Koya, Crack detection on concrete surfaces using V-shaped features., World Comput. Sci. Inf. Technol. J. 8 (1) (2018).

[9] M.R. Halfawy, J. Hengmeechai, Automated defect detection in sewer closed circuit television images using histograms of oriented gradients and support vector machine, Autom. Constr. 38 (2014) 1–13.

[10] S.I. Hassan, L.M. Dang, I. Mehmood, S. Im, C. Choi, J. Kang, Y.-S. Park, H. Moon, Underground sewer pipe condition assessment based on convolutional neural networks, Autom. Constr. 106 (2019) 102849.

[11] C. Oh, L.M. Dang, D. Han, H. Moon, Robust sewer defect detection with text analysis based on deep learning, IEEE Access 10 (2022) 46224–46237.

[12] Y. Li, H. Wang, L.M. Dang, H.-K. Song, H. Moon, Vision-based defect inspection and condition assessment for sewer pipes: A comprehensive survey, Sensors 22 (7) (2022) 2722.

[13] L.M. Dang, H. Wang, Y. Li, L.Q. Nguyen, T.N. Nguyen, H.-K. Song, H. Moon, Lightweight pixel-level semantic segmentation and analysis for sewer defects using deep learning, Constr. Build. Mater. 371 (2023) 130792.

[14] Y. Jiang, D. Pang, C. Li, A deep learning approach for fast detection and classification of concrete damage, Autom. Constr. 128 (2021) 103785.

[15] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, M. Pietikäinen, Deep learning for generic object detection: A survey, Int. J. Comput. Vis. 128 (2) (2020) 261–318.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.

[17] J. Hosang, R. Benenson, B. Schiele, Learning non-maximum suppression, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4507–4515.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-end object detection with transformers, in: European Conference on Computer Vision, Springer, 2020, pp. 213–229.

[20] H. Wan, L. Gao, Z. Yuan, H. Qu, Q. Sun, H. Cheng, R. Wang, A novel transformer model for surface damage detection and cognition of concrete bridges, Expert Syst. Appl. (2022) 119019.

[21] G. Zhang, Y. Pan, L. Zhang, Deep learning for detecting building façade elements from images considering prior knowledge, Autom. Constr. 133 (2022) 104016.

[22] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, J. Dai, Deformable detr: Deformable transformers for end-to-end object detection, 2020, arXiv preprint arXiv:2010.04159.

[23] C.V. Dung, et al., Autonomous concrete crack detection using deep fully convolutional neural network, Autom. Constr. 99 (2019) 52–58.

[24] J.K. Chow, Z. Su, J. Wu, P.S. Tan, X. Mao, Y.-H. Wang, Anomaly detection of defects on concrete structures with the convolutional autoencoder, Adv. Eng. Inform. 45 (2020) 101105.

[25] S.E. Park, S.-H. Eem, H. Jeon, Concrete crack detection and quantification using deep learning and structured light, Constr. Build. Mater. 252 (2020) 119096.

[26] S. Teng, Z. Liu, X. Li, Improved YOLOv3-based bridge surface defect detection by combining high-and low-resolution feature images, Buildings 12 (8) (2022) 1225.

[27] Y. Dong, J. Wang, Z. Wang, X. Zhang, Y. Gao, Q. Sui, P. Jiang, A deep-learning-based multiple defect detection method for tunnel lining damages, IEEE Access 7 (2019) 182643–182657.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: European Conference on Computer Vision, Springer, 2014, pp. 740–755.

[29] Y. Yu, B. Samali, M. Rashidi, M. Mohammadi, T.N. Nguyen, G. Zhang, Vision-based concrete crack detection using a hybrid framework considering noise effect, J. Build. Eng. 61 (2022) 105246.

[30] M. Veluchamy, B. Subramani, Image contrast and color enhancement using adaptive gamma correction and histogram equalization, Optik 183 (2019) 329–337.

[31] H. Wu, Y. Qu, S. Lin, J. Zhou, R. Qiao, Z. Zhang, Y. Xie, L. Ma, Contrastive learning for compact single image dehazing, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 10551–10560.

[32] Y. Li, S. Li, H. Du, L. Chen, D. Zhang, Y. Li, YOLO-ACN: Focusing on small target and occluded object detection, IEEE Access 8 (2020) 227288–227303.

[33] L. Ziyin, Z.T. Wang, M. Ueda, Laprop: a better way to combine momentum with adaptive gradient, 2020, arXiv preprint arXiv:2002.04839.

[34] B. Xu, N. Wang, T. Chen, M. Li, Empirical evaluation of rectified activations in convolutional network, 2015, arXiv preprint arXiv:1505.00853.

[35] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, S. Savarese, Generalized intersection over union: A metric and a loss for bounding box regression, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 658–666.

[36] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren, Distance-iou loss: Faster and better learning for bounding box regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, No. 07, 2020, pp. 12993–13000.

[37] D. Minh, H.X. Wang, Y.F. Li, T.N. Nguyen, Explainable artificial intelligence: a comprehensive review, Artif. Intell. Rev. 55 (5) (2022) 3503–3568.

[38] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, Ssd: Single shot multibox detector, in: European Conference on Computer Vision, Springer, 2016, pp. 21–37.

[39] F. Zhou, H. Zhao, Z. Nie, Safety helmet detection based on YOLOv5, in: 2021 IEEE International Conference on Power Electronics, Computer Applications, ICPECA, IEEE, 2021, pp. 6–11.

[40] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2016) 1137–1149.

[41] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2961–2969.

[42] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, Q. Tian, Centernet: Keypoint triplets for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6569–6578.

[43] S. Moradi, T. Zayed, Real-time defect detection in sewer closed circuit television inspection videos, in: Pipelines 2017, 2017, pp. 295–307.