Automated Aging Building Defect Recognition and Analysis Using Transformers

L. Minh Dang[1]; Muhammad Fayaz[2]; Quoc Bao To, Ph.D.[3]; Gayoon Lee, Ph.D.[4]; Hyoung-Kyu Song[5]; Kihak Lee[6]; and Hyeonjoon Moon[7]

Abstract: Aging infrastructure poses significant safety risks due to the potential for undetected structural defects leading to catastrophic failures. Traditional manual inspection approaches are considered labor-intensive, time-consuming, and susceptible to human error. We propose a transformer-based defect detection framework specifically designed for detecting defects in aging buildings to solve the existing issues of manual methods. Transformer-based models rely on self-attention mechanisms to dynamically focus on relevant image regions. This approach avoids the limitations of conventional convolutional networks by capturing long-range contextual dependencies and adapting them to variations in defect appearance. To enhance the model's generalizability and adaptability to real-world scenarios, a data preprocessing module is proposed to handle varying lighting conditions, angles, and environmental factors. Moreover, the proposed model integrates novel classification loss and matching cost to the state-of-the-art detection transformer (DETR) with the improved denoising anchor boxes (DINO) model to improve its performance and stabilize the learning process for accurate defect detection. The ability to analyze and visualize attention weights from multiscale features of the framework promotes transparency and builds confidence in automated defect detection systems. The experimental results reveal that the AD-TR framework outperforms competitive deep learning–-based object detection models with an average mean average precision (mAP) of 83.1%. **DOI: 10.1061/JCCEE5.CPENG-6797.**

Introduction

Defect inspection in aging buildings is crucial due to the growing demand for safe and sustainable living environments in modern cities. Many older structures, built decades ago, have deteriorated over time. Structure deterioration can lead to issues like structural deficiencies, safety hazards, and energy inefficiencies (Faqih and Zayed 2021). Regular inspections are critical for identifying defects, such as cracks, water infiltration, and foundation instability, which can severely affect the safety and well-being of residents. In densely populated regions like Korea, where urban renewal and redevelopment are pressing concerns, early defect detection plays a vital role in mitigating costly repairs, improving property values, and supporting long-term sustainability goals (Tulbure et al. 2022).

Traditionally, defect inspections relied on manual assessments by trained inspectors, who visually evaluate structures for signs of wear, structural damage, and other issues (de Brito et al. 2020). While effective, this method was labor-intensive, time-consuming, and prone to human error, which could lead to failure to detect subtle defects or inconsistent judgments. To address these drawbacks, machine learning (ML) algorithms were adopted to improve the inspection process by automating the analysis of visual and sensor data (Valero et al. 2019). Compared with the manual approach, these models not only increased efficiency and consistency but also enabled the rapid processing of large volumes of inspection data. However, traditional ML models often depend on laborious feature engineering and struggle to generalize across diverse defect types or environmental conditions, such as varying lighting or material textures (Taye 2023). These limitations highlight the need for more robust, adaptive solutions to meet the evolving demands of building inspection.

Advancements in deep learning (DL) have transformed defect inspection by providing scalable, high-performance alternatives to conventional ML models. DL models automatically extract hierarchical features from large-scale data to deliver state-of-the-art results without the need for manual feature engineering (Xiao et al. 2020). These capabilities have led to DL's growing adoption in structural inspection tasks, including classification, segmentation, and detection. For defect classification, convolutional neural networks (CNNs) demonstrated state-of-the-art performance in identifying structural issues such as cracks, moisture damage, and corrosion. In defect segmentation, architectures like U-Net and DeepLab (Dang et al. 2023a; Wang et al. 2021) enabled pixel-level localization of damaged regions. Defect detection, which identifies the precise location of defects within an image, received increasing attention due to its practical relevance in inspection workflows

[1]Researcher, Institute of Research and Development, Duy Tan Univ., Da Nang 550000, Viet Nam; Professor, Faculty of Information Technology, Duy Tan Univ., Da Nang 550000, Viet Nam; Research Professor, Dept. of Information and Communication Engineering and Convergence Engineering for Intelligent Drone, Sejong Univ., Seoul 05006, Republic of Korea.

[2]Ph.D. Candidate, Dept. of Computer Science and Engineering, Sejong Univ., Seoul 05006, Republic of Korea.

[3]Professor, Deep Learning Architecture Research Center, Dept. of Architectural Engineering, Sejong Univ., 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea.

[4]Professor, Deep Learning Architecture Research Center, Dept. of Architectural Engineering, Sejong Univ., 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea. ORCID: https://orcid.org/0000-0001-6824-8019

[5]Professor, Dept. of Information and Communication Engineering and Convergence Engineering for Intelligent Drone, Sejong Univ., Seoul 05006, Republic of Korea. ORCID: https://orcid.org/0000-0002-3274-4982

[6]Professor, Deep Learning Architecture Research Center, Dept. of Architectural Engineering, Sejong Univ., 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea.

[7]Professor, Dept. of Computer Science and Engineering, Sejong Univ., Seoul 05006, Republic of Korea (corresponding author). Email: hmoon@sejong.ac.kr

Note. This manuscript was submitted on January 14, 2025; approved on October 24, 2025; published online on March 18, 2026. Discussion period open until August 18, 2026; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801.

(Li et al. 2022c). Models such as faster region-based CNN (Faster R-CNN) and You Only Look Once (YOLO) excel at detecting various types of defects (Gao et al. 2019; Lee et al. 2020; Li et al. 2022a). Recent advancements have also introduced anchor-free methods, such as fully convolutional one-stage object detector (FCOS) (Tian et al. 2020) and CenterNet (Duan et al. 2019), that mitigate the need for manually tuned anchor boxes and improve adaptability to variable defect scales. Furthermore, advanced CNN architectures incorporate multiscale feature aggregation via Feature Pyramid Networks (FPN) (Lin et al. 2017a) and attention-based modules (e.g., in hybrid CNN-transformer models) to better capture global contexts beyond traditional local receptive fields (Carion et al. 2020; Dong et al. 2022). In practice, this means that modern detectors leverage deeper feature maps and attention layers so that even distant image regions can influence defect prediction.

Despite enhancements, certain standard CNN architectures, particularly those without extensive global modeling mechanisms, still face challenges in civil engineering applications. For instance, they may struggle to fully capture long, irregular defect patterns, such as defects that span across images and require understanding of nonlocal continuity and contextual relationships (e.g., distinguishing interconnected defect segments from isolated noise or shadows) (Akinosho et al. 2020; Tulbure et al. 2022). In fact, studies of crack segmentation note that CNNs "primarily rely on local neighborhood information, limiting their effectiveness in modeling global context" (Shi et al. 2025). This is because local receptive fields, even with multiscale aggregation, prioritize nearby pixel interactions and may overlook broader structural dependencies that are critical for defects to form complex, global patterns (Guo et al. 2022). Empirical evidence from recent studies shows that while data augmentation (e.g., scaling, rotation) can improve handling of irregularity, they do not inherently address the need for explicit global reasoning, leading to reduced performance on data sets with highly variable defect morphologies. For example, Mundt et al. (2019) showed that CNN models achieved lower performance on complex, nonlocal patterns like fine and elongated cracks compared with more uniform defects. Recent defect-detection architectures use cross-attention or vision-transformer backbones to explicitly capture image-wide dependencies (Huang et al. 2024a). Therefore, long-range relationship modeling is critical in civil applications because a detector might see only fragmented pieces of a defect and miss the entire structure. By modeling a global context, our approach aims to better integrate these contextual cues that standard CNNs may overlook.

The attention mechanism from transformer architectures, originally introduced for natural language processing (NLP) (Han et al. 2023), was recently adapted to computer vision (CV) tasks, particularly object detection. The Detection Transformer (DETR) (Carion et al. 2020) replaced hand-designed proposal pipelines of CNN-based approaches with an end-to-end transformer framework. Although DETR achieved results comparable [43.6% average precision (AP)] to those of the previous classical CNN-based detectors on the common objects in context (COCO) benchmark, they suffered from slow training convergence of 500 epochs. Zhu et al. (2020) argued that DETR's slow convergence was due to the instability of Hungarian matching and the decoder cross-attention. Deformable-DETR (Zhu et al. 2020) mitigated these issues with deformable attention. Deformable DETR converged at roughly 300 epochs and achieved a higher AP of 45.5 on the COCO benchmark. These advances have encouraged application of transformers in civil-engineering inspection. For example, Dang et al. (2023b) adopted a Deformable-DETR backbone for concrete-defect detection (four-class data set), while Guo et al. (2023) proposed a so-called crack transformer built on a Swin transformer encoder-decoder for pavement analysis. Huang et al. (2024b) noted that early DETR-like models are not well suited to tiny objects because object-query positional priors are not customized for such scales. Subsequent improvements, such as DN-DETR's denoising training (Li et al. 2022b), reportedly achieved 46.0 AP in only 12 epochs by reducing the difficulty of bipartite graph matching. DINO (Zhang et al. 2022b) further improves stability and accuracy via better denoising, anchor-box denoising, and refined positional adjustments. Experimental results demonstrated that DINO achieved substantial improvements of $+6.0AP$ compared with DN-DETR using the same 12-epoch setting.

Despite many advances, standard DETR-like variants still suffer from instability in the bipartite matching step during training, which can produce inconsistent assignment of predictions to ground truth objects. This instability often leads to inconsistent training dynamics, suboptimal alignment between predicted and target boxes, and sensitivity to noisy or ambiguous object proposals, all of which degrade detection accuracy and robustness in complex scenes (Liu et al. 2023). These problems are especially consequential for aging building inspection, where common defects, such as hairline cracks, peeling paint, spalling, falling debris, fence damage, and window failures, span a broad range of scales, shapes, and textures and are frequently small and co-located with clutter.

We present an improved transformer-based framework for aging building defect detection based on the DINO backbone with a refined matching mechanism designed to produce more consistent, reliable assignments and faster convergence. Our model is trained on a large-scale data set of 406,000 images covering seven defect categories typical of aging buildings. To increase interpretability, we extract and visualize mean attention weight maps from the decoder's final layers to reveal which features the model uses for localization and classification. Together, these improvements yield a more accurate and robust detector that is better suited for practical structural health monitoring and large-scale urban inspection tasks.

The section "Aging Building Defect Data" describes the large-scale aging building defect data. "Overall Framework" outlines the main components of the proposed framework. "Customized DINO: AD-TR" provides a detailed explanation of each part of the AD-TR model. "Experimental Results" introduces and evaluates the proposed model through a series of experiments. The final sections "Discussion" and "Conclusions and Future Work" explain key findings and their contributions, present conclusions, and discuss potential future work.

Aging Building Defect Data

Existing data sets on building defects are small in both size and scope. For example, Perez et al. trained a defect inspection model using a data set of 2,622 images containing only three defect types: mold, stain, and paint deterioration (Perez et al. 2019). Similarly, Wang et al. targeted efflorescence and spalling detection in historic masonry using only 500 images (Wang et al. 2019). In contrast, we utilize a large-scale, diverse data set, which contains approximately 406,954 images of seven common defect classes. This data set not only addresses the scarcity of annotated examples in earlier studies but also enhances generalization by capturing broader variations in defect morphology, lighting conditions, and surface textures.

The data were collected by the National Information Society Agency of Korea (NIA) as part of practical inspection initiatives aimed at improving the robustness and real-world applicability of defect detection in aging buildings. Data collection was led by the Mediagroup Human & Forest in collaboration with key industry partners. Notable contributors included HCI+ for on-site data

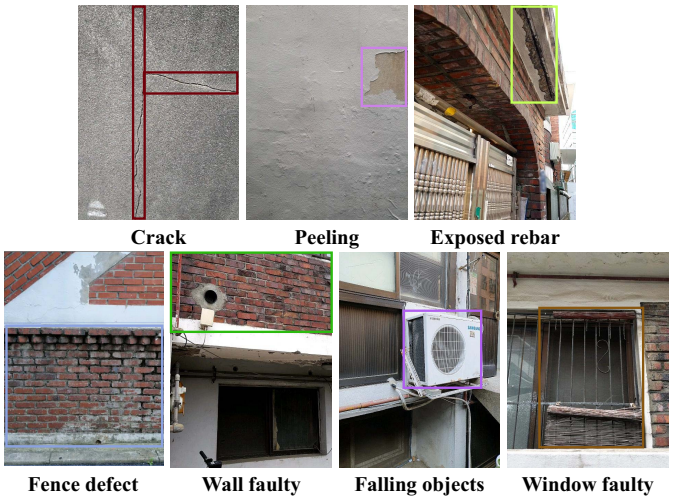

Fig. 1. Seven aging building defects from the data set. The bounding boxes pinpoint the sample defect for each class.

collection, and Infiniq, which managedfor data refinement and processing tasks.

Images were captured by smartphones to document five structure categories over 20 years old in 25 districts in Seoul, South Korea: single-family homes, multifamily buildings, row houses, apartments, and nonresidential buildings. The data set included seven defect types—cracks, peeling, exposed rebar, fence defect, wall faulty, falling objects, and window faulty—selected for their prevalence in aging buildings and alignment with South Korean regulatory standards outlined in the Enforcement Decree of the Building Act (Articles 9-2, 23-2, 23-7) (Korea 2008). They were further validated by studies on apartment defect taxonomy (Kim et al. 2022) and are widely recognized in Korea for assessing structural safety. Examples of each defect type are presented in Fig. 1.

The seven aging building defect classes are defined as follows:

- *Crack*: linear fractures on concrete or wall surfaces, caused by foundation settlement, structural movement, or external stress factors such as thermal expansion, seismic activity, or vibration-induced fatigue.
- *Peeling*: localized loss of adhesion between surface materials (e.g., paint, plaster) and the substrate, driven by hydrological stress, inadequate bonding, or substandard initial application
- *Exposed rebar*: visible protrusion of reinforcing steel (rebar) due to concrete cover deterioration, often caused by aging, environmental exposure, or poor construction. This exposes critical structural elements to corrosion and poses significant safety risks.
- *Fence defect*: structural failure or physical damage to fencing systems (e.g., misalignment, breakage) caused by environmental wear, improper installation, or mechanical impact. This creates safety hazards and compromises aesthetic quality
- *Wall faulty*: nonuniform deformation (e.g., bulging, bowing, or leaning) through the wall thickness, triggered by foundation instability, moisture intrusion, thermal stress, or poor construction practices. These defects undermine load-bearing capacity and stability.
- *Falling objects*: gravity-induced risks from inadequately anchored architectural elements, such as heating, ventilation, and air conditioning (HVAC) units, façade components or loose debris, which threaten pedestrian safety and exacerbate structural degradation.
- *Window faulty*: Compromised integrity of window systems due to fractured frames, degraded seals, or missing hardware, which affects insulation, security, water infiltration, and safety of the building.

Fig. 2 describes the total number of annotated images for each defect class. In total, 406,954 images and their corresponding labeling files were collected. Of this number, 80% (325,530) were assigned as the training set. Among these, 25% (65,106) were used as the validation data. The remaining 20% of the original data set (81,424) were designated as the testing set to assess the generalization capability and performance of the model on unseen data.

Overall Framework

Fig. 3 shows the main components of the proposed framework for detecting and analyzing defects in aging buildings, referred to as AD-TR. In this abbreviation, "AD" stands for aging building defect detection; "TR" highlights the transformer-based defect recognition approach. Each component is broadly described as follows:

- *Data preprocessing*: raw images captured by smartphones often contain artifacts such as low-light conditions, uneven illumination, or motion blur. The preprocessing stage improves image quality for robust defect detection. Here, two state-of-the-art preprocessing methods are implemented: URetinex-Net (Wu et al. 2022) for low-light correction and a deep Wiener deconvolution network (Dong et al. 2020) for denoising. Then comes standard data augmentation, including flipping, rotating, cropping, and brightness/contrast adjustment, to diversify the training data set and improve model generalization.
- *Defect detection*: An end-to-end transformer-based framework for detecting structural defects in aging buildings utilizing the DINO architecture, a state-of-the-art DETR variant. To improve model performance, two components are introduced: (1) a positional-aware loss function that injects positional metrics into DETR's classification loss to penalize spatial misalignments between predicted and ground truth (GT) defects; and (2) an enhanced matching cost that refines the Hungarian matching algorithm by integrating geometric priors to improve alignment between predicted bounding boxes and irregular defect shapes. These adaptations address critical challenges in aging infrastructure (e.g., scale variability, sparse defect distributions), which enable robust detection of complex, real-world structural anomalies.
- *Defect analysis*: This component analyzes the identified defects and evaluates potential risks. Because DINO is transformer-based, it computes self-attention weights that quantify how much each image patch attends to every other patch during inference. We extract the attention weight matrixes from all heads in the final transformer layer, average them to produce a single attention score per patch, and bilinearly upsample these scores back to the original image resolution to form a spatial heatmap. The map provides interpretable insights into the defect detection process by highlighting the regions that most influenced the model's predictions. In practice, the attention map enables engineers to improve performance by suggesting additional viewpoints where attention is weak.

Methodology

Data Preprocessing

Earlier studies showed that artifacts and noise in images significantly degrade the performance of defect inspection frameworks (Xu et al. 2023; Dang et al. 2023a). As a result, preprocessing is essential to enhance image quality and improve defect detection for robust structural analysis in aging buildings. We implemented two complementary preprocessing models: URetinex-Net (Wu et al. 2022) for low-light enhancement and deep Wiener

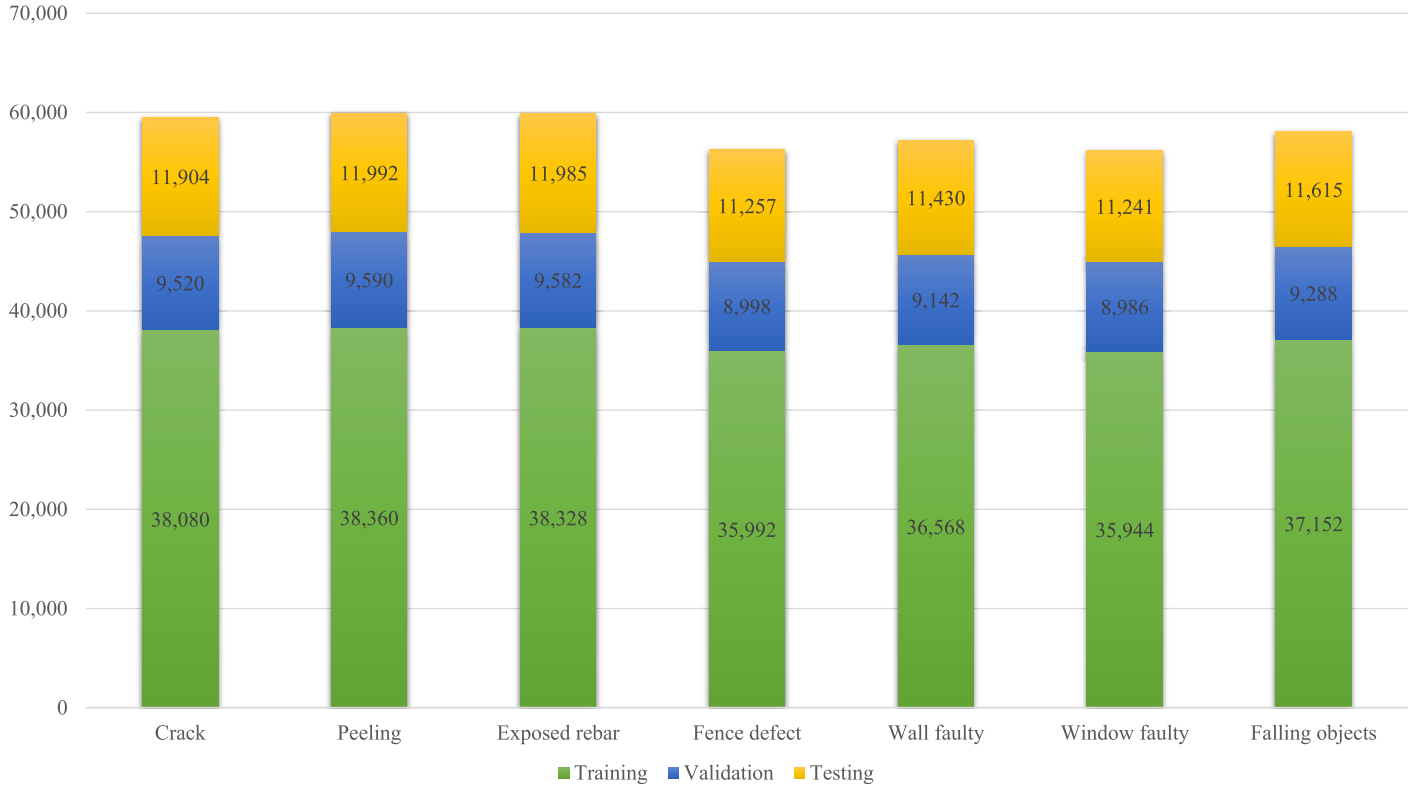

Fig. 2. Total number of annotated images for each defect type.

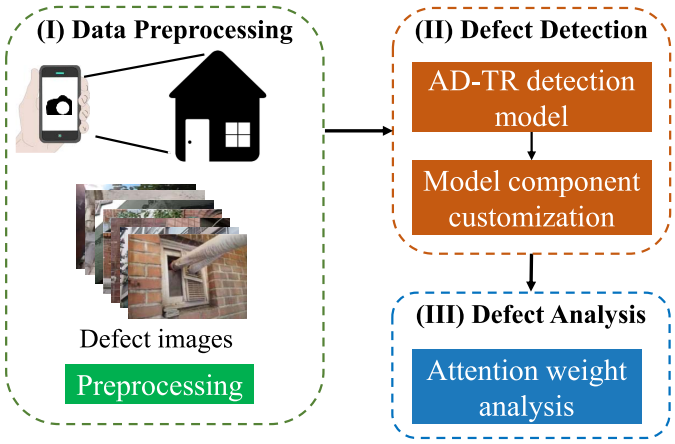

Fig. 3. Key processes in the AD-TR framework.

deconvolution (Dong et al. 2020) for denoising. While both are DL-based and are less computationally efficient than traditional methods, their superior ability to preserve and restore details critical for defect detection justifies their use in this safety-critical application.

URetinex-Net was selected for its ability to effectively enhance low-light images while preserving critical structural details necessary for detecting subtle defects. Motivated by the Retinex theory, URetinex-Net decomposes images into reflectance and illumination layers using a deep unfolding network that integrates traditional optimization techniques with data-driven learning (Wu et al. 2022). Unlike histogram equalization and gamma correction, which often amplify noise and introduce artifacts, URetinex-Net adaptively suppresses artifacts while enhancing details through its optimization and illumination adjustment modules. This is particularly advantageous in our application, where low-light conditions obscure cracks, surface deterioration, and the like. The original research demonstrated its superiority over state-of-the-art low-light enhancement methods in terms of quantitative image quality metrics.

Next, the output from URetinex-Net was processed using the deep Wiener deconvolution model to address blurriness and improve defect clarity. Unlike traditional deblurring methods such as Gaussian filtering, which may oversmooth images and lose high-frequency information, this approach performs deconvolution in a deep feature space learned by neural networks. The combination of Wiener deconvolution and DL refines details progressively and reduces artifacts via a multiscale feature refinement module. Dong et al. (2020) showed that this method outperforms traditional as well as other DL-based deblurring techniques in detail preservation, which made it well-suited for the crucial task of preserving structural details.

Fig. 4 illustrates the effectiveness of our preprocessing pipeline by comparing raw input images (low-light, blurred, poorly illuminated) with their enhanced counterparts. The combination of URetinex-Net and deep Wiener deconvolution improves image clarity, contrast, and defect visibility. For example, cracks that are difficult to detect in a low-light image become much more visible after preprocessing, as do critical details such as peeling paint and exposed surfaces. In the case of blurred imagery, the preprocessing pipeline effectively reduces blur and sharpens edges, allowing hairline cracks and early-stage structural degradation to become clear. Moreover, preprocessing introduces no unintended distortions or quality degradation in regions unaffected by lighting or blur issues.

DETR with Improved Denoising Anchor Boxes

The DETR model (Carion et al. 2020) redefines object detection as a direct set prediction task using a Transformer encoder-decoder

Fig. 4. Visual comparison of random images before and after applying the preprocessing module for normal conditions, blurriness, low light, and illumination.

architecture. In DETR, an image is first encoded via a convolutional backbone and positional embeddings; next a Transformer encoder captures global contextual relationships. A fixed number of learnable object queries are fed into the decoder, which attends to encoder outputs to simultaneously predict bounding boxes and class labels. A bipartite matching loss enforces a unique one-to-one correspondence between predicted queries and GT objects, enabling end-to-end training without hand-made components like anchor generation or non-maximum suppression. While DETR establishes a clean, unified detection paradigm, it suffers from slow convergence and relatively coarse localization, particularly for small or densely packed objects.

DINO is an advanced end-to-end object detection model that builds on the DETR framework (Zhang et al. 2022b). It addresses key limitations of earlier DETR-based models, such as slow convergence and suboptimal localization accuracy, through innovations that enhance both training efficiency and detection precision. These improvements make DINO a robust choice for detecting defects in aging infrastructure, where accurate localization and classification are critical. Similar to the baseline DETR model, DINO includes a backbone, a Transformer encoder, and a decoder. It is built on DETR by applying: (1) denoising training (Li et al. 2022b), which accelerates convergence by injecting noisy GT boxes during training; (2) dynamic anchor boxes (Liu et al. 2022), which replace static anchors with adaptive four-dimensional (4D) reference boxes (x, y, w, h) for dynamic localization adjustments; and (3) deformable attention (Zhu et al. 2020), which reduces computational complexity by sampling sparse key points around reference positions. DINO also introduces three new methods:

- To improve one-to-one matching, DINO introduces contrastive denoising training by simultaneously integrating both negative and positive samples into the GT. Noise is added to a GT box in two different ways: labeling the box with less noise as positive and the one with more noise as negative. Contrastive denoising training allows the model to distinguish correct and incorrect detections more effectively.
- DINO implements a mixed query selection method, where initial anchor boxes are extracted from encoder outputs as position queries. This improves initialization of the queries and leads to better detection performance.
- DINO proposes "look forward twice," which adjusts the parameters updated by gradients from the back layer. To enhance the accuracy of the overall pyramid structure, it optimizes the adjacent front layer by utilizing the refined box information from the back layer.

A complete workflow of the transformer-based aging buildings defect detection framework is illustrated in Fig. 5.

Contrastive Denoising

Denoising training was first introduced in DN-DETR (Li et al. 2022b) by adding controlled noise to GT bounding boxes during training. This generated two types of samples: positive (slightly perturbed boxes) and negative (heavily displaced boxes). This approach mitigated duplicate predictions and improved anchor selection by explicitly teaching the model to distinguish between overlapping or noisy proposals. However, DN-DETR struggled to correctly predict anchors having no nearby objects ("no object"). This limitation led to increased false positives and reduced precision, particularly in complex scenarios with dense object arrangements.

Contrastive denoising (CDN) (Nam et al. 2024) extended denoising by explicitly distinguishing between objects and background regions while further refining anchor selection. Unlike traditional denoising methods that focused solely on reconstructing GT boxes from perturbed inputs, CDN introduces both positive and negative samples during training by adding different levels of noise to GT bounding boxes. Detailed descriptions of positive and negative samples follows.

- *Positive samples*: generated by applying a small noise level λ_1 to GT boxes. Slightly perturbed, these boxes remain close to the actual object locations to encourage the model to learn precise localization and classification.
- *Negative samples*: created by applying a larger noise level λ_2 to the GT boxes. The generated boxes are significantly displaced from any actual object. They are trained to predict "no object", explicitly teaching the model to suppress false positives in background regions.

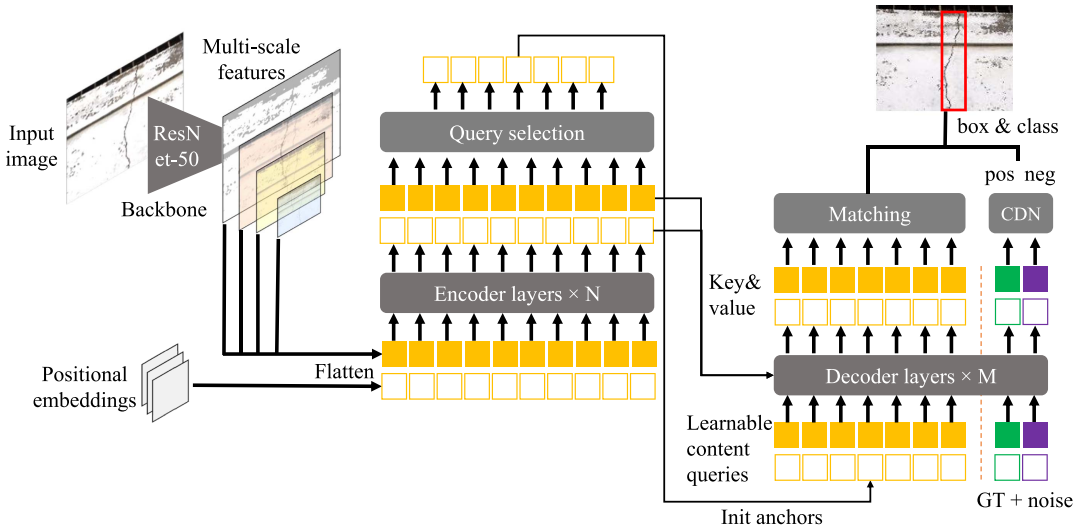

Fig. 5. Workflow of the framework.

The contrastive approach optimized training efficiency by guiding the model toward better anchor predictions while minimizing false positives. By carefully selecting.

With noise levels that keep the noise for negative samples small enough to produce hard negatives but larger than that for positive samples, CDN enabled the model to handle challenging negative samples. These hard negatives were more difficult for the model to classify correctly, but were highly beneficial for improving overall detection performance. Moreover, CDN addressed the issue of duplicate predictions by improving anchor selection. Training on both positive and negative samples helped the model associate each anchor with a unique object or correctly identify it as background.

Mixed Query Selection

The Deformable-DETR employed a dynamic query selection mechanism that utilized top-K encoder features to initialize both content and positional queries for object detection. Specifically, it generated object proposals from encoder outputs and used them to initialize the queries. However, the selected features were preliminary content features that did not receive further processing or refinement. The lack of additional processing introduced misinterpretation and ambiguity into the decoder's operation. As a result, it could affect the model's ability to accurately capture complex object representations (Zhang et al. 2022b). Therefore, a novel MQS strategy was introduced in the DINO framework to initialize anchors as positional queries for the decoder.

MQS stabilized DINO's decoder by keeping its content queries fixed but allowing its positional queries to adapt during training. In this way, rich image features learned early on by the model were not distorted. At the same time, the model dynamically adjusted the positional queries to different object placements and scenarios to adapt its spatial understanding. Thus, DINO maintained stable content representations while dynamically adjusting to varying object positions by separating the learning of content and positional queries (Zhang et al. 2022b).

"Look Forward Twice" Box Prediction

In traditional Transformer-based object detection models like Deformable-DETR, "look forward once" was commonly used during training (Zhu et al. 2020). In this approach, the gradients from one decoder layer did not affect the learning of subsequent layers, which enabled stable learning and mitigated issues like gradient explosion or vanishing during back-propagation. However, "look forward once" slowed down the overall learning process because each layer was learned in isolation and did not exploit shared information across layers.

DINO introduced "look forward twice," which accelerated convergence by letting each decoder layer benefit from the box refinements of previous layers. Instead of isolating updates layer by layer as in "look forward once," it fed updated box predictions forward twice so that later layers received more accurate priors. This extra information flow sped up training and boosted box accuracy without compromising gradient stability.

For a given input bounding box b_{i-1} at the $(i-1)$th layer of the Transformer decoder, the final predicted bounding box $b_i^{(\text{pred})}$ at the ith layer is derived through sequential updating. The process involves computing the box offset, updating the bounding box, and detaching gradients to prevent certain back-propagation paths

$$\Delta b_i = \text{Layer}_i(b_{i-1}), \qquad b_i' = \text{Update}(b_{i-1}, \Delta b_i) \tag{1a}$$

$$b_i = \text{Detach}(b_i'), \qquad b_i^{(\text{pred})} = \text{Update}(b_{i-1}', \Delta b_i) \tag{1b}$$

where b_i' = undetached version of b_i; and b_i = detached version of b_i', which means that gradients are not back-propagated through b_i during training (detachment is crucial to prevent instability in the gradient flow caused by the recursive use of updated boxes); $b_i^{(\text{pred})}$ = final predicted bounding box at layer i. The term $\text{Update}(\cdot,\cdot)$ is a function responsible for refining the box b_{i-1} using the predicted box offset Δb_i. Moreover, the same box updating as in Deformable-DETR is implemented (Zhu et al. 2020).

Customized DINO: AD-TR Model

DETR Loss and Matching Cost

Most variations of DETR models used similar loss functions and matching designs derived from Deformable-DETR. While other DETR-like detectors might introduce minor modifications, the core principles of their loss functions and matching strategies remained largely consistent (Zhang et al. 2022a). The DETR loss function comprised three main components:

- *Classification Loss* (L_{cls}): measures the error in classifying each predicted bounding box.

- Box L1 *Loss* (L_{bbox}): computes the L1 distance between predicted and GT bounding box coordinates.
- *Generalized intersection over union loss* (L_{GIoU}): evaluates the overlap between predicted and GT bounding boxes to facilitate a scale-invariant measure.

The box L1 and GIoU losses were used for object localization and remained unmodified from the original Deformable-DETR. The primary customization in DINO was classification loss, where it employed Focal Loss to address class imbalance and focus training on hard examples. Focal loss modified the standard cross-entropy loss by introducing a modulating factor to focus learning on hard-to-classify examples. The classification loss in DINO was defined as

$$L_{\text{cls}} = \sum_{i=1}^{N_{\text{pos}}} (1 - p_i)^{\gamma} \text{BCE}(p_i, 1) + \sum_{i=1}^{N_{\text{neg}}} p_i^{\gamma} \text{BCE}(p_i, 0) \quad (2)$$

where N_{pos} and N_{neg} = numbers of positive and negative examples; BCE = binary cross-entropy loss; p_i = prediction probability for the ith example; and γ = hyperparameter for focal loss.

The matching process was a crucial step in training object detection models, where predictions were assigned to GT objects to determine positive and negative examples for loss computation. Each GT object typically matched only one prediction, which was considered a positive example. Predictions that were not matched to any GT were treated as negative examples. To perform this matching, a cost matrix $C \in \mathbb{R}^{N_{\text{pred}} \times N_{\text{gt}}}$ was computed, where N_{pred} and N_{gt} represent the number of predictions and GT instances, respectively. The Hungarian matching algorithm was then applied to the cost matrix to find the optimal one-to-one assignment between predictions and GT objects that minimized the total cost. The assignment ensured that each GT object was matched to a unique prediction.

The final loss function focused on the classification cost, box L1 cost C_{bbox}, and GIoU cost C_{GIoU} (Liu et al. 2023). The classification cost for the ith prediction and the jth GT is computed as follows:

$$C_{\text{cls}}(i, j) = |1 - p_i|^{\gamma} \text{BCE}(p_i, 1) + p_i^{\gamma} \text{BCE}(1 - p_i, 1) \quad (3)$$

The formula is a modified version of the focal loss cost. Focal loss emphasizes forcing positive examples to be predicted as 1 by adding an additional penalty term to the classification cost.

Position-Supervised Loss

Object detection models face a multiobjective optimization challenge, where they not only need to classify objects correctly but also need to localize them precisely within an image. The DINO model implements a position-aware score to the classification loss function to supervise the prediction probabilities of positive examples based on their localization accuracy. Previous studies have shown that the classification loss of DINO is modified to integrate the intersection over union (IoU) between the predicted bounding boxes and the GT. The modified classification loss is defined as

$$L_{\text{cls}}^{(\text{new})} = \sum_{i=1}^{N_{\text{pos}}} [f_1(s_i) - p_i]^{\gamma} \text{BCE}(p_i, f_1(s_i)) + \sum_{i=1}^{N_{\text{neg}}} p_i^{\gamma} \text{BCE}(p_i, 0) \quad (4)$$

where N_{pos} and N_{neg} = number of positive and negative examples; s_i = intersection over union (IoU) or other metrics between the GT and the predicted bounding box for the ith example; and γ = hyperparameter for focal losses. It is challenging to use IoU directly in the loss function because it can introduce instability due to its small magnitude (typically between 0 and 1). The function $f_i(s_i)$ maps s_i to a target label in the range [0, 1], which can take different forms such as linear mapping s_i, squared mapping s_i^2, or exponential mapping e^{s_i}.

Liu et al. (2023) reported that $f_1(s_i) = \varepsilon(s_i^2)$ was a simple, smooth way to amplify differences between high- and low-quality matches. Unlike an exponential, s^2 remains numerically stable and forces the model to focus on refining the predicted accurate boxes and broaden the effective range of the localization loss. Moreover, its derivative $2s$ integrates neatly into the BCE gradients by imposing little extra computational cost, leading to more robust training than purely linear weighting. The ε is a normalization parameter to preserve a stable dynamic range for the positional weights. It rescales each set of s_i^2 within a training example so that the largest value is either matched to the maximum IoU among all candidate pairs or simply set to 1.0.

Position-Modulated Matching

The position-supervised classification loss aims to encourage predictions with high IoU scores but low classification scores. Position-modulated matching (PMC) is introduced to inject spatial quality directly into the Hungarian matching cost. Rather than treat classification confidence and localization quality as independent terms, classification confidence is modulated by a learned function of a positional score. If a prediction i has classification probability p_i and positional metric s_i' (rescaled GIoU), s_i' is transformed via a tunable function $f_2(s_i') = \sqrt{s_i'}$ to obtain a weight in [0, 1]. Liu et al. (2023) proved that this function achieved the highest performance because it softened the harsh penalty on medium-quality boxes and prevented the model from being too confident in inaccurate predictions. The new pair matching cost is as follows:

$$C_{\text{cls}}^{(\text{new})}(i, j) = |1 - p_i f_2(s_{i'})|^{\gamma} \text{BCE}(p_i f_2(s_i'), 1) - (p_i f_2(s_i'))^{\gamma} \text{BCE}(1 - p_i f_2(s_i'), 1) \quad (5)$$

The term $C_{\text{cls}}^{(\text{new})}$ down-weights predictions whose boxes have low overlap with the target. For example, if s_i' is small, the term $p_i f_2(s_i')$ shrinks, which makes the classification contribution to the matching cost less dominant. The exponent γ (as in focal loss) further focuses the matching on uncertain/borderline examples, while BCE terms encourage the chosen pairing to maximize true positives and penalize false positives.

The approach addresses the multiobjective optimization challenge in object detection by indirectly reflecting the matching costs in the training objectives. The intuition behind PMC is twofold. First, the model is discouraged from pairing ground truth objects with predictions that are either spatially inaccurate or overconfident in the wrong place. Second, PMC ensures that only predictions with both high spatial overlap and high classification achieve the lowest matching cost.

Transformer Attention Weights Visualization

The attention weights extracted from the decoder's last layers provide insights into how the model prioritizes different regions of the input image during object detection. These weights are crucial for interpreting the model's decision-making, as they quantify the importance assigned to each spatial location by individual attention heads (Parmar et al. 2018). By projecting these weights onto the input image, the regions where the model places more attention when predicting bounding boxes and class labels can be visualized. Regions with higher attention weights are highlighted to demonstrate the model's ability to focus on relevant objects within the image. The attention weights W are computed by applying the softmax function to a scaled dot-product attention scores matrix A

$$W = \text{softmax}\left(\frac{A}{\sqrt{d_k}}\right) \tag{6}$$

where d_k = dimensionality of the key vectors. This scaling factor mitigates the risk of excessively large dot products, which could otherwise saturate the softmax gradients. The attention scores matrix A, computed as the dot product between the queries Q and the transpose of the keys K, captures the raw compatibility between input positions

$$A = QK^T \tag{7}$$

Each element w_{ij} in W represents the weight assigned to the jth position of the input feature map when processing the ith query. These weights determine the influence of that region on the corresponding output prediction. For visualization, the initial attention weights matrix in the form of a square matrix of dimensions $[H \times W, H \times W]$ is reshaped into a tensor of shape $[H, W, H, W]$, where H and W are the height and width of the feature map. This transformation aligns the attention weights with the spatial structure of the input to generate attention maps that highlight the spatial regions the model attends to for each query.

The generated attention maps provide valuable insights into the model's detection mechanisms and its robustness in various object detection tasks. For instance, they reveal whether the model prioritizes salient object features (e.g., edges, textures, or distinctive parts) during detection. Visualization also exposes potential shortcomings, such as the model attending to irrelevant regions or failing to capture important details. By diagnosing these patterns, researchers can refine parameters, adjust training strategies, or enhance data augmentation pipelines to improve performance.

Experimental Results

A series of experiments was designed to evaluate the AD-TR model's performance in different settings using aging building defect data. We begin by defining the evaluation metrics that quantify detection accuracy, precision, and robustness. We also outline the hardware and software configuration used for implementing the AD-TR model and baseline detection models. To ensure a fair and reproducible evaluation, a detailed explanation of the hyperparameters configured for both the proposed model and the existing models is provided.

The experimental findings are organized into five key categories. In particular, the impact of the proposed preprocessing pipeline on defect detection performance is analyzed. Also, the impact of PSL and PMC on the DINO model is determined through ablation studies and qualitative and quantitative evaluations of AD-TR on the seven defect types, including challenging scenarios, are provided. The performance of AD-TR is compared against other standard detection models. Finally, attention weights are visualized to reveal how the model prioritizes critical defect regions.

Evaluation Metrics

Three fundamental components of the confusion matrix—true positives (TP), false negatives (FN), and false positives (FP)—are applied to evaluate the effectiveness of the proposed defect detection model. These components are crucial for calculating mean average precision (mAP), which comprehensively assesses performance. Precision and recall provide a more comprehensive analysis of the model's performance. These metrics help identify the trade-offs between detecting as many defects as possible (high recall) and minimizing false detections (high precision).

Precision measures the proportion of correctly identified defects among all detections. A high precision value indicates that a model has a low false-positive rate, which indicates that it rarely misidentifies nondefect regions as defects

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

Recall quantifies the proportion of actual defects that are correctly detected. A high recall value indicates that a model identifies almost all actual defects in the images. It is expressed as

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

In our study, the primary evaluation metric is the mAP, which measures all defect types in the data set. The mAP is calculated based on the IoU metric, with a threshold set at 0.5, which means that a predicted bounding box is considered a correct detection if its IoU with the corresponding GT bounding box is higher than or equal to 0.5.

$$mAP = \frac{1}{N_{\text{classes}}} \sum_i AP_i \tag{10}$$

where N_{classes} = number of defect types; and AP_i = average precision for a defect class i, which is calculated by integrating the precision-recall curve for i.

Implementation

The aging building defect detection framework is implemented and trained using PyTorch 2.0, a Python DL library. All model implementations are carried out using MMDetection, an open-source object detection toolbox providing a unified platform that supports numerous state-of-the-art detection algorithms and network architectures. ResNet-50 (Wu et al. 2019) is used as the default backbone to ensure the consistency and reliability of all experiments. The model was originally trained on the ImageNet data set because of its rich feature representations learned from a large and diverse set of images. Training and testing were conducted on an NVIDIA A100 GPU with a batch size of 16.

For all Transformer-based experiments (AD-TR and DETR variants), models were trained for 15 epochs, with validation performed at the end of each one. Early stopping was implemented based on validation bounding box loss using a patience of 3 epochs and a minimum improvement threshold of $\Delta = 10^{-3}$. Training terminated if the validation loss failed to improve by at least Δ for 3 consecutive epochs, with weights restored from the epoch achieving the lowest validation loss.

In our experimental setup, the hyperparameters are consistent with those used in DINO (Zhang et al. 2022b). In particular, AD-TR contains a 6-layer Transformer encoder and a 6-layer Transformer decoder, each with a hidden feature dimension of 256. This configuration balances model complexity and computational efficiency to enable effective learning of feature representations. Nine hundred learnable object queries are set for AD-TR. Multiple patterns helps the model capture diverse object representations and improves detection performance. The learning rate begins at 1×10^{-4} and utilizes a simple learning rate scheduler. For training schedules of 15 epochs with the ResNet-50 backbone, the learning rate is reduced by a factor of 0.1 at the 11th epoch. We employed the AdamW optimizer with a weight decay of 1×10^{-4} to prevent overfitting by penalizing large weights.

For the baseline DINO model, we use a combination of the L1 loss and the GIoU loss for bounding box regression, and the focal

Table 1. Preprocessing effect on performance of various detection models

Approach	Model	mAP (%)	Precision (%)	Recall (%)
Without preprocessing	YOLOv5 (Wang et al. 2023)	66.7	68.2	66.5
	SSD (Liu et al. 2016)	60.1	62.5	61.3
	Faster R-CNN (Ren et al. 2017)	63.3	62.9	61.0
	SOLO (Wang et al. 2020)	72.6	70.4	70.7
	DETR (Carion et al. 2020)	69.2	68.0	67.3
	DINO (Zhang et al. 2022b)	71.5	70.9	72.6
	AD-TR (ours)	**75.4**	**77.2**	**74.7**
With preprocessing	YOLOv5 (Wang et al. 2023)	73.2	74.5	71.9
	SSD (Liu et al. 2016)	67.3	68.7	66.1
	Faster R-CNN (Ren et al. 2017)	68.7	68.4	67.8
	SOLO (Wang et al. 2020)	74.1	72.5	72.8
	DETR (Carion et al. 2020)	72.6	73.8	73.4
	DINO (Zhang et al. 2022b)	77.2	78.0	76.7
	AD-TR (ours)	**83.1**	**84.6**	**83.9**

Note: Bold indicates the model with highest performance.

loss for classification. The L1 term encourages the predicted boxes to align precisely with the GT coordinates, while the GIoU loss addresses the limitations of standard IoU in nonoverlapping cases by directly optimizing spatial overlap and maintaining scale invariance (Carion et al. 2020). For classification loss, the focal loss mitigates class imbalance by down-weighting dominant negative samples and focusing on hard examples (Lin et al. 2017b), with hyperparameter $\alpha = 0.25$ for reducing background weight and $\gamma = 2$ to suppress gradients from well-classified instances. For AD-TR, PLS is integrated into the classification loss with a weight of 6.0, which is an optimal value reported by Liu et al. (2023). The classification cost is modulated by the square root of the normalized GIoU [$f_2(s) = \sqrt{s}$] and this term is multiplied by a cost weight of 2 in the Hungarian matching algorithm. Finally, as in DETR (Carion et al. 2020), we apply auxiliary losses after each decoder layer to accelerate convergence by providing intermediate supervision. Similar to Deformable-DETR (Zhu et al. 2020), we also include additional intermediate losses after the query selection module.

Defect Identification Performance Assessment

Preprocessing Performance

As discussed in the section "Data Preprocessing," the preprocessing steps include low-light enhancement, deblurring, and image enhancement, to improve the visual quality of images, which are often affected by factors like poor lighting and environmental degradation in the context of aging buildings. Here we provide a comprehensive evaluation of the impact of the proposed preprocessing on the performance of the AD-TR model and six other state-of-the-art object detection models. These models include the Single Shot MultiBox Detector (SSD) (Liu et al. 2016), YOLOv5 (Wang et al. 2023), Faster R-CNN (Ren et al. 2017), Segmenting Objects by Locations (SOLO) (Wang et al. 2020), Detection Transformer (DETR) (Carion et al. 2020), and DINO (Zhang et al. 2022b). The mAP values reported in Table 1 are derived from a single run of each model with the same fixed random seed, both with and without the preprocessing module.

The results reported in Table 1 reveal the impact of the preprocessing module on the performance of various DL-based object detection models. Without preprocessing, all detection models achieved moderate performance, with mAP values ranging from 60.1% to 75.4%. In particular, the AD-TR framework outperformed all other models in this category with a mAP of 75.4%. The SOLO model achieved the next highest mAP of 72.6%, followed by the DINO baseline at 71.5%, and DETR at 69.2%. The results indicate that even without preprocessing, the AD-TR model exhibits superior capability in detecting defects compared with the other models.

After performing preprocessing, all models showed significant improvements in their detection results. The AD-TR model showed the highest improvements, with its mAP increasing from 75.4% to 83.1%, precision from 77.2% to 84.6%, and recall from 74.7% to 83.9%. The improvement of 7.7% in mAP highlights the effectiveness of preprocessing in improving defect detection. Other models also benefited substantially from the preprocessing module. For example, DINO's mAP improved from 71.5% to 77.2%, and SOLO's mAP increased from 72.6% to 74.1%. On average, the preprocessing module contributed to an increase of approximately 5.3% in mAP for all models.

The results prove the prepreocessing pipeline's critical role in improving detection. It is particularly effective for images affected by challenging conditions such as poor lighting, blurriness, and low contrast, which are common in outdoor environments and aging infrastructure. The models extract more meaningful features by enhancing image quality before training. The substantial improvements in various state-of-the-art models highlight the effectiveness of the proposed preprocessing module in improving the performance of defect detection systems.

Ablation Study

Figs. 6(a–d) show representative failure cases of the baseline DINO detector on our aging building data set, annotated with arrows for clarity. The principal failure modes are

- *Missed detections*: Fig. 6(a) shows a long hairline crack that DINO fails to detect (arrow). Hairline, high-frequency cracks are often missed or returned as fragmented detections. Similarly, Fig. 6(d) shows that tiny defects (falling objects) whose closely matches the surrounding wall are missed.
- *Background confusion*: Fig. 6(b) presents a textured brickwork where the model confuses weathering or texture for a peeling defect (yellow arrow).
- *Occlusion and clutter*: Figs. 6(b–d) (orange arrows) emphasize common failures of the baseline DINO caused by large, low-contrast regions or complex scene clutter of nearby structures. These conditions tend to produce fragmented, oversegmented, or low-confidence detections.

Collectively, these examples indicate that the defects in this study are frequently small, textural, and often occur against cluttered backgrounds. Such properties encourage DINO's query-based matcher to favor compact, high-confidence boxes and thus produce

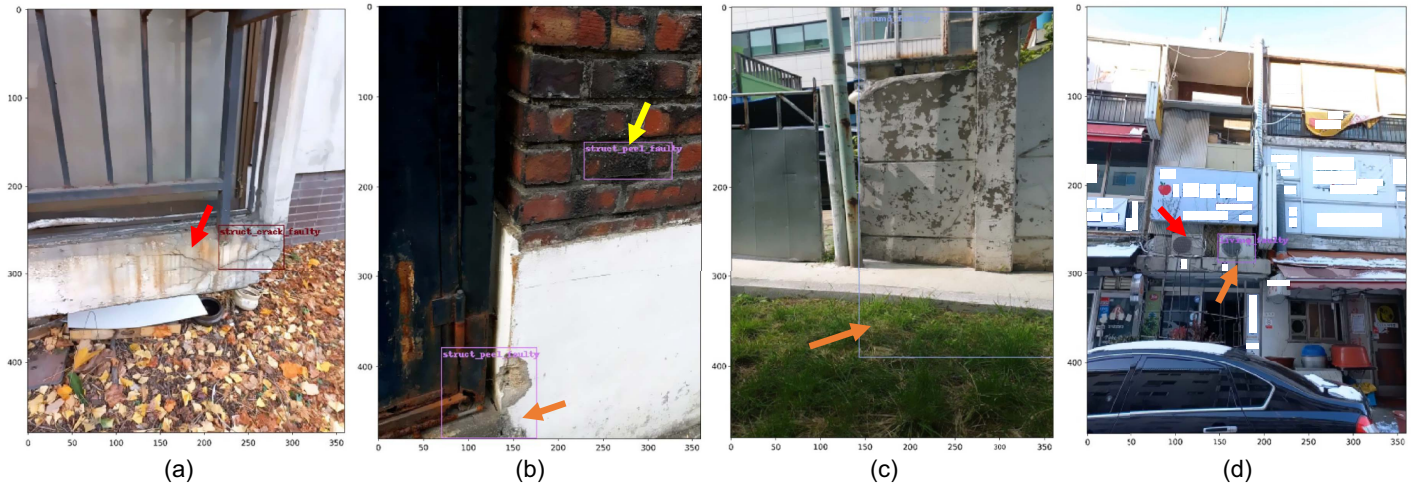

Fig. 6. Typical qualitative failure cases of DINO detector on the aging building test set: (a) thin hairline crack; (b) small-object; (c) background confusion; and (d) occlusion/clutter. Panels (a–d) show typical failure modes, where arrows point to the errors.

errors. This behavior motivates the use of PSL to train classification scores to reflect localization quality and PMC, which biases the matching cost toward well-localized candidates. PSL and PMC aim to improve detection stability for thin, low-contrast, or occluded defects.

Table 2 illustrates the effectiveness of the proposed components in the AD-TR model through a series of ablation studies. The AD-TR (DINO baseline) model, which does not include PSL or PMC, achieves an initial mAP of 77.2%. The significant improvement of 80.9% in detection is seen in AD-TR (1) after introducing PSL into the model. It indicates that adding a PSL helps the model better learn the spatial relationships and positional information critical for accurate defect detection. On the other hand, AD-TR (2), which implements the PMC module alone achieves a 76.4% mAP, which is worse than the AD-TR (DINO baseline). Liu et al. (2023) and Cai et al. (2024) demonstrated that without PSL the classification scores may not be aligned with positional metrics, so PMC's modulation can be noisy or unstable. AD-TR (3), which integrates both PSL and PMC, leads to the highest mAP of 83.1% among all configurations tested. The addition of PMC refines the matching between predicted bounding boxes and GT by modulating the cost with positional information. In general, the improvements from 77.2% to 80.9%, and finally to 83.1% highlight the cumulative benefits of adding both the PSL and PMC into the AD-TR model. These results validate the effectiveness of the proposed modules in improving the model's ability to detect and localize defects in aging buildings.

Fig. 7 is a comparison of the DINO baseline and the proposed AD-TR model, which integrates PSL and PMC modules. Figs. 7(a and b) highlight missed detections: red arrows mark thin, hairline cracks and tiny defects that DINO either ignores or returns as fragmented boxes, whereas AD-TR correctly detects them. Fig. 7(c) shows background confusion (yellow arrow), where DINO mistakenly detects fliers on the body of a utility pole as a defect; AD-TR suppresses this false positive. Finally, Figs. 7(d and e) show failures from occlusion and clutter (orange arrows). In the presence of low-contrast regions or complex scene clutter, DINO's predictions become oversegmented but AD-TR's predictions are more precise.

These examples highlight two important points about the domain and AD-TR's improvements. First, aging building defects are sometimes small, highly textural, and complex. Therefore, methods that favor compact, high-confidence box hypotheses like DINO may produce fragmented localization or generate false-positive predictions. Second, AD-TR appears to improve both localization and false-positive reduction. PSL and PMC modules encourage classification scores to reflect localization quality and biases matching toward well-localized candidates.

Table 2. Ablations for different AD-TR model settings

Model	PSL	PMC	mAP
AD-TR (DINO-baseline)	—	—	77.2
AD-TR (1)	X	—	80.9
AD-TR (2)	—	X	76.4
AD-TR (3)	X	X	83.1

Note: PSL = position-supervised loss; and PMC = position-modulated cost in matching.

AD-TR Performance Evaluations

Fig. 8 shows the outputs of the AD-TR model for the seven defect classes. The model consistently demonstrates a high level of accuracy in identifying defects for all categories. Notably, the model performed exceptionally well in identifying instances of the peeling defect by effectively capturing even the subtle variations in texture that signify early signs of material degradation. In the case of falling objects and exposed rebar, AD-TR shows remarkable precision in identifying defects even when they are small or situated at a significant distance from the camera. Overall, the results demonstrate the AD-TR model's effectiveness and reliability in real-world applications, such as proactive maintenance and safety interventions in aging infrastructure.

Fig. 9 shows AD-TR's robust performance in multidefect scenarios by effectively identifying multiple aged building structural issues within a single image. In Fig. 9(a), it correctly identifies both a vertical structural crack and a faulty window defect; in Fig. 9(b), it distinguishes a small patch of peeling from a crack at the base of the wall; in Fig. 9(c) it correctly detects two adjacent falling object defects and an extended horizontal crack below them. By using the attention mechanisms, AD-TR successfully captures global contextual relationships, which enable it to precisely predict defects even when they overlap or coexist in cluttered environments. The results demonstrate that AD-TR is well-suited for real-world automated inspections because it effectively handles the complex, multidefect scenarios commonly encountered in practice.

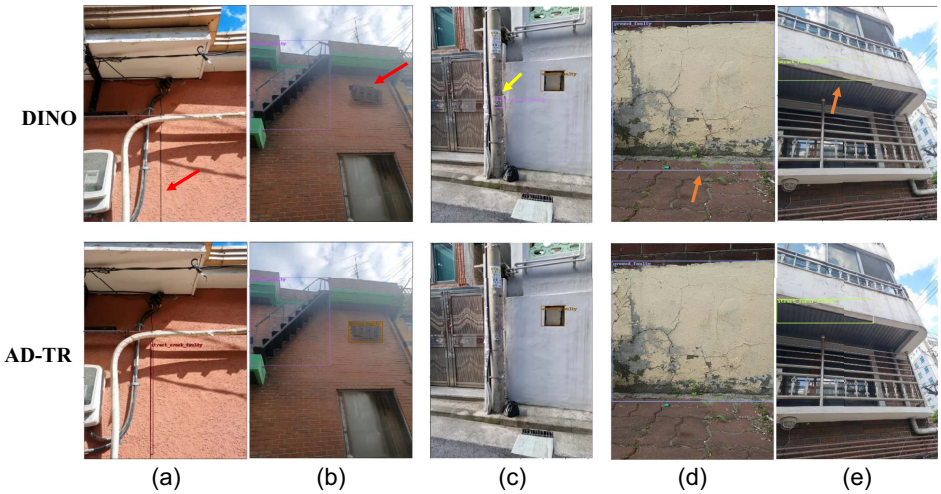

Fig. 7. Qualitative comparison of predictions between the DINO baseline and the proposed AD-TR model. (a and b) missed detections, (c) background confusion, and (d and e) occlusion and cluster. Arrows point to the errors.

Fig. 8. Bounding box predictions by the AD-TR model for each defect type: (a) peeling; (b) crack; (c) falling objects; (d) exposed rebar; (e) window faulty; (f) wall faulty; and (g) fence defect.

Table 3 compares the detection performance of two models, DINO and AD-TR, on the test set using 5-fold cross-validation. Metrics include mAP0.5, precision, and recall, with standard deviations indicating variability across different folds. Overall, AD-TR consistently outperforms the DINO baseline on all defect categories with an average gain of roughly 6%–7% in mAP@0.5. For instance, in the "Window faulty" class, AD-TR achieves an mAP of 90.3% (±1.6), significantly outperforming DINO's 82.4% (±1.9). Similarly, AD-TR shows higher precision (89.0% versus 81.2%) and recall (91.5% versus 82.9%) for this defect, which highlights its improved ability to detect true positives while minimizing false alarms.

In addition, AD-TR maintains relatively low standard deviations, suggesting greater stability and robustness during cross-validation. The performance metrics in Table 3 highlight the effectiveness of AD-TR in detecting various defect types. However, both models struggle with the "Wall faulty" class, where AD-TR's mAP (72.8% ±1.8) still falls behind other defects, indicating potential challenges in detecting this specific anomaly. A potential explanation for the poor detection of Wall faulty defects lies in their low visual contrast with the surrounding wall surface, particularly when they exhibit colors or textures similar to the background. For example, dirt accumulation, climbing vines, or weathering effects (e.g., water stains, discoloration) can create patterns that overlap with structural

Fig. 9. AD-TR model performance in multidefect scenarios: (a) crack and window faulty; (b) peeling and crack; and (c) crack and falling objects.

Table 3. 5-fold cross-validation of AD-TR model performance for each defect class evaluated using

Model	Defect class	mAP0.5 (%)	Precision (%)	Recall (%)
DINO	Window faulty	82.4 ± 1.9	81.2 ± 1.6	82.9 ± 1.7
	Falling objects	80.6 ± 1.6	80.9 ± 1.3	81.4 ± 1.5
	Exposed rebar	79.8 ± 1.2	78.5 ± 1.7	76.6 ± 1.9
	Crack	77.5 ± 1.3	77.8 ± 1.1	75.1 ± 1.4
	Peeling	73.2 ± 2.1	74.9 ± 1.7	72.3 ± 1.3
	Fence defect	71.4 ± 1.8	75.7 ± 1.6	73.1 ± 1.2
	Wall faulty	69.2 ± 1.5	71.5 ± 1.4	68.6 ± 1.1
AD-TR	Window faulty	90.3 ± 1.6	89.0 ± 1.4	91.5 ± 0.9
	Falling objects	88.7 ± 1.2	88.9 ± 1.2	89.4 ± 1.1
	Crack	86.3 ± 1.4	86.5 ± 1.5	84.1 ± 1.7
	Exposed rebar	85.1 ± 1.5	84.2 ± 1.8	83.6 ± 1.6
	Fence defect	80.2 ± 1.7	80.5 ± 1.8	79.3 ± 1.4
	Peeling	78.4 ± 1.3	81.2 ± 1.6	80.5 ± 1.3
	Wall faulty	72.8 ± 1.8	78.9 ± 1.9	76.5 ± 1.9

defects like cracks or peeling. This semantic ambiguity, where nondefect anomalies mimic the appearance of true defects, leads to misclassification. In addition, the model may struggle to distinguish "Wall faulty" from co-occurring anomalies (e.g., cracks with peeling paint), as their overlapping visual traits and textural similarities reduce discriminative feature learning.

Fig. 10 shows four cases showing the AD-TR model's performance on challenging detection tasks. Figs. 10(a and b) highlight the model's robustness in successfully detecting defects in challenging scenarios. In Fig. 10(a), the model accurately identifies a crack in the wall, despite the presence of distracting graffiti and complex textures. In Fig. 10(b), it successfully detects a faulty window in a cluttered scene with multiple pipes and wires, despite the window being distant and viewed from a nonoptimal angle. Therefore, these two cases demonstrate the model's ability to precisely detect small or partially obscured defects in complex environments by isolating relevant features from background noise.

However, Figs. 10(c and d) reveal areas where the model's accuracy could be enhanced. In Fig. 10(c), although the model correctly detects two exposed rebar defects, it fails to detect an air conditioning unit (indicated by the red arrow). The results reveal that the model struggles with identifying defects associated with appliances or external equipment that blend into the environment, particularly when they are not integral parts of the building's structural components. In Fig. 10(d), the red arrow points to a false

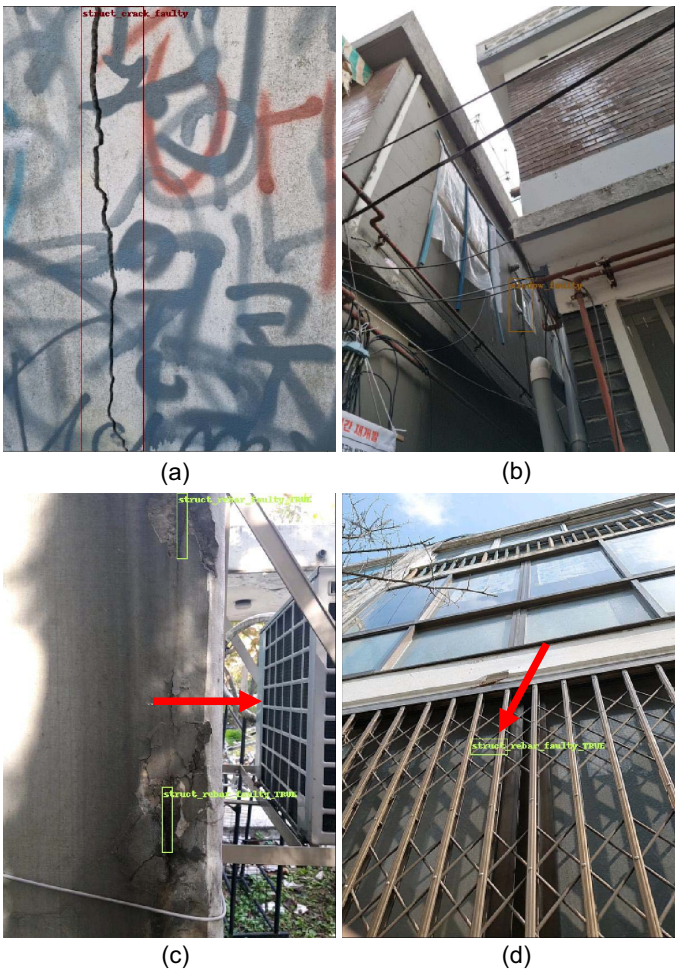

Fig. 10. AD-TR model outputs for hard cases: (a) wall with graffiti and complex textures; (b) cluttered scene with multiple pipes and wires; (c) defects associated with appliances or external equipment blending into the environment; and (d) visually similar linear metal structures resembling defects.

positive where the model incorrectly classifies a window bar as an exposed rebar. The misclassification highlights the challenge of distinguishing between visually similar linear metal structures when they share visual characteristics common to defects like exposed

Table 4. Comparison of AD-TR model and other detection models

Model	mAP (%)	Precision (%)	Recall (%)	Speed (ms/image)
YOLOv5 (Wang et al. 2023)	73.2	74.5	71.9	12
SSD (Liu et al. 2016)	67.3	68.7	66.1	15
Faster-RCNN (Ren et al. 2017)	68.7	68.4	67.8	85
SOLO (Wang et al. 2020)	74.1	72.5	72.8	80
DETR (Carion et al. 2020)	72.6	73.8	73.4	120
Deformable-DETR (Zhu et al. 2020)	74.5	75.2	74.8	88
DINO (Zhang et al. 2022b)	77.2	78	76.7	113
AD-TR (ours)	83.1	84.6	83.9	180

rebar. These examples indicate that the model could benefit from further training on a wider variety of defect types and from incorporating contextual cues to better differentiate between objects with similar visual features.

Comparison for AD-TR

The performance comparison between the proposed AD-TR model and several state-of-the-art models on the testing data set are provided in Table 4. The AD-TR model significantly outperforms all other models with the highest mAP of 83.1%. For example, it surpasses DINO, the next best model (Zhang et al. 2022b), which achieves a mAP of 77.2%, and standard models like YOLOv5 (Wang et al. 2023) and SSD (Liu et al. 2016), which achieve mAPs of 73.2% and 67.3%. In terms of precision and recall, AD-TR also leads with 84.6% precision and 83.9% recall. The recorded metrics indicate their effectiveness in accurately detecting defects and their improvement over existing approaches.

Inference speed is a critical factor in real-world applicability. Although the AD-TR model processes images at an average speed of 117 ms per image, it is competitive given its high accuracy. Models like YOLOv5 and SSD offer faster processing times of 12 and 15 ms per image but at a cost of lower mAP scores. Similarly, Transformer-based models such as DETR (Carion et al. 2020) and DINO have inference speeds of 120 and 113 ms per image, and they obtain lower accuracy compared with AD-TR. The AD-TR model shows a good balance between performance and speed, which is suitable for applications where high detection accuracy is essential and real-time processing is less critical.

Our two-stage preprocessing pipeline—URetinex-Net for low-light enhancement followed by deep Wiener deconvolution for denoising—introduces an additional time of approximately 63 ms per image (25 ms for URetinex-Net + 18 ms for deep Wiener deconvolution) when executed on an NVIDIA A100 GPU (Taiwan, China) (batch size = 1). This translates to a 12% increase in total inference time (from 117 to 180 ms per image) but a 7.7% mAP improvement (Table 4). We consider these costs acceptable for infrastructure inspection scenarios, where enhanced detection reliability outweighs the moderate increase in processing time.

Fig. 11 compares the detection performance of three Transformer-based models: AD-TR, DETR, and DINO. Compared with the other models, AD-TR (green line) consistently achieves the highest mAP throughout training. DINO (orange line) starts with a slightly higher mAP than DETR (blue line) and maintains steady improvement. In contrast, DETR shows unstable performance initially, with a sharp drop around the sixth epoch before gradually improving toward the end. Overall, AD-TR not only reaches a higher final mAP but also shows stable and efficient learning progression, which makes it more effective and reliable for defect detection tasks in this comparison.

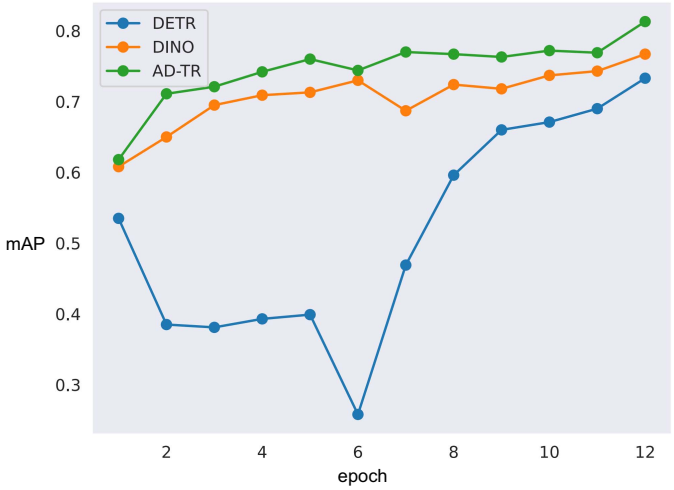

Fig. 11. Comparison of performance among transformer-based models: proposed AD-TR, DETR (Carion et al. 2020), and DINO (Zhang et al. 2022b); evaluation metric is mAP.

Small Object Analysis

We adopt COCO-style evaluation and construct a small object subset for the defect classes Crack, Peeling, Exposed rebar, and Window faulty, where small instances are common. Following COCO, an instance is considered small if its ground truth bounding box area falls in the small area bins (in px^2)

$$A_{\text{bbox}} < 32^2 = 1{,}024 \text{ px}^2 \tag{11}$$

We report AP_S by restricting evaluation to ground truth instances in this bin, which matches the COCO definition of the "small" category. As shown in Table 5, 22.9% of training instances from the four defect classes are small objects, with proportions ranging from 18.6% (Exposed rebar) to 27.9% (Peeling). AD-TR consistently outperforms Deformable-DETR on AP_S for the four defect classes. Absolute gains are 6.1 (Peeling: 22.3 to 28.4), 6.2 (Crack: 20.7 to 26.9), 4.2 (Exposed rebar: 18.1 to 22.3), and 4.1 (Window faulty: 21.5 to 25.6). This leads to an overall AP_S improvement of 5.2 (from 20.6 to 25.8). These results demonstrate AD-TR's effectiveness in detecting small defect instances, particularly for Crack and Peeling classes with the highest prevalence of small objects.

Attention Weight Analysis Results

The attention weight analysis visualized in Fig. 12 highlights the AD-TR model's ability to focus on relevant features of various structural defects through Transformer-based attention mechanisms. For example, the model accurately places high attention weights along the crack's path for the crack defect. The attention weight map effectively reveals the defect's thin linear structure despite surrounding textures. Similarly, for window faulty and exposed rebar defects, high Transformer attention weights are recorded for areas where structural irregularities or exposed materials are visible.

For more complex scenarios, such as falling objects, peeling, and fence defects, the attention maps reveal that the model can identify broader defect regions, even in the presence of cluttered backgrounds and overlapping elements. For instance, in the "falling objects" case, the model focuses on objects like the air conditioner unit and nearby pipes, which could potentially pose a structural risk. In the peeling and fence defect cases, the attention weights are extended to all areas that show degradation or texture changes. These examples demonstrate the AD-TR model's ability to adapt its attention to capture both localized and scattered defect patterns.

Table 5. Comparison of per-category COCO metrics for small defect classes between AD-TR and Deformable-DETR models

Class	Total images	Small instances	Deformable-DETR-AP_S	AD-TR-AP_S
Peeling	38,360	10,702 (27.9%)	22.3	28.4
Crack	38,080	8,796 (23.1%)	20.7	26.9
Exposed rebar	38,328	7,129 (18.6%)	18.1	22.3
Window faulty	35,944	7,871 (21.9%)	21.5	25.6
Overall improvement	150,712	34,498 (22.9%)	20.6	25.8

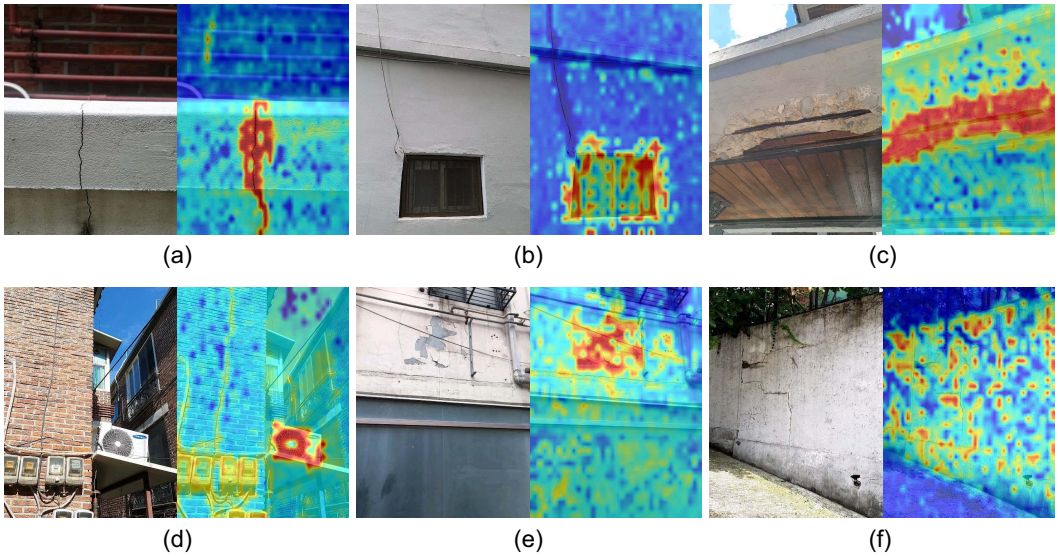

Fig. 12. Sample visualization of the attention weight analysis for various defects: (a) crack; (b) window faulty; (c) exposed rebar; (d) falling objects; (e) peeling; and (f) fence defect.

Discussion

AD-TR Model

The AD-TR model is a Transformer-based object detector specifically customized for defects in aging buildings. Built on top of DINO, AD-TR introduces two targeted components, PSL and PMC, which explicitly utilize positional quality metrics during training. PSL modifies classification supervision so that positive labels are scaled by a positional quality (IoU), encouraging class scores to reflect localization fidelity. PMC injects positional information directly into the Hungarian matching cost, biasing assignments toward well-localized predictions. These additions help the model learn spatial relationship and positional cues that are critical for reliably detecting structural defects that vary widely in size, shape, and appearance, and they reduce sensitivity to background clutter and complex scene elements.

Empirically, AD-TR demonstrates strong performance across extensive experiments. On the aging building defect data set, it achieved an mAP of 83.1%, with precision and recall of 84.6% and 83.9%, outperforming a range of competitive baselines including CNN detectors (YOLOv5, SSD, Faster R-CNN, SOLO) and Transformer variants (DETR, Deformable-DETR, DINO). We find that integrating PSL/PMC into classification loss and matching cost significantly improves robustness in failure modes where the DINO baseline struggles, such as missed detections, background confusion, and occlusion/clutter (Section "Ablation Study") for qualitative examples. Finally, AD-TR achieves an inference time of approximately 117 ms per image. Therefore, it offers a practical balance between accuracy and efficiency for large-scale inspection tasks.

In addition, the AD-TR model offers practical benefits for structural health monitoring and maintenance of aging buildings. The model's generalizability is significantly improved through data preprocessing, which makes it suitable for deployment in practical applications. The integration of advanced Transformer architecture and optimization allows the FD-TR model to scale efficiently to larger data sets and higher-resolution images. Moreover, explainability is crucial for gaining users' trust in advanced technological solutions. The AD-TR framework is capable of extracting and visualizing attention weights from multiscale features to enhance analysis and understanding of the model's outputs. Finally, its strong performance and adaptability make it a promising tool for automating the detection and management of building defects.

Real-Life Defect Detection

While AD-TR demonstrates increased performance for all evaluation metrics in all seven defect categories, it is crucial to prioritize recall over other metrics particularly in the context of safety-related applications. Missing a structural defect (false negative) can lead to catastrophic outcomes, whereas misclassifying a nondefect as a defect (false positive) poses comparatively lower risk, as such errors can be resolved through subsequent verification. With preprocessing, the DINO baseline achieves precision of 78.0% and recall of 76.7% while AD-TR (PSL + PMC) achieves precision of 84.6% and recall of 83.9%. The improvements are stable across 5-fold cross-validation as per-class standard deviations are 1%–2% while AD-TR's class gains are generally 6%–7% (Table 3). This increase means that for every 1,000 actual defects the system would miss about 233 with the baseline compared with around 161 with

AD-TR—approximately 72 fewer missed defects (31% reduction). This translates to about 3.6 fewer missed defects per 1,000 images at a low prevalence of 50 defects/1,000 images, and around 21.6 fewer missed defects per 1,000 images at a higher prevalence of 300 defects/1,000 images. In conclusion, although AD-TR (PSL + PMC) increases recall from 76.7% to 83.9%, the number of predicted positives requiring human verification rises about 1%; crucially, this gain corresponds to 72 fewer missed defects per 1,000 images.

Conclusions and Future Work

We have introduced the AD-TR model, a Transformer-based object detection framework for detecting defects in aging buildings in Korea. The framework is trained on a data set containing 406,954 images of seven defect types. By introducing additional components to the state-of-the-art DINO models, AD-TR shows stabilized learning with higher detection performance. The proposed preprocessing module improves the model's robustness and effectiveness in identifying a variety of defects under challenging conditions, such as poor lighting and complex backgrounds. Extensive experiments demonstrated that the AD-TR model, with the highest mAP of 83.1%, outperforms several state-of-the-art detection models.

The practical implications of AD-TR are significant for structural health monitoring. Its good generalizability and adaptability make it suitable for diverse real-world scenarios. Although on-site inspection is still required to capture images of a structure, our model drastically shortens the overall process by allowing engineers to take and upload images rather than perform detailed manual assessments on the spot. By accurately detecting defects early, the model contributes to extending the life of infrastructure and ensuring safety.

Despite its strengths, this study has several limitations that should be noted. First, the data set images were collected with a limited range of smartphone models. Variations in camera resolution, lens distortion, and image compression may impact detection performance in practice. Second, uncontrolled factors such as extreme lighting, occlusions, and motion blur can degrade image quality and thus affect model accuracy. Third, while AD-TR excels at localizing and classifying defect types, it does not yet quantify the severity of each defect (e.g., depth of cracks or degree of spalling).

Future work will focus on expanding the data set to include a wider variety of imaging devices and real-world shooting scenarios to improve robustness across smartphone brands and environmental conditions. We also plan to integrate a severity estimation component, which will enable the system to rank defects by urgency. Finally, we aim to optimize the model's computational efficiency for on-device inference and to explore its applicability to other infrastructure types, such as bridges and tunnels.

Data Availability Statement

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (RS-2024-00461244) and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) under the metaverse support program to nurture the best talents (IITP-2024-RS-2023-00254529) through a grant funded by the Korea government [Ministry of Science and ICT (MSIT)].

Author Contributions

L. Minh Dang: Conceptualization; Methodology; Writing – original draft; Writing – review and editing. Muhammad Fayaz: Methodology; Writing – review and editing. Quoc Bao To: Investigation; Visualization. Gayoon Lee: Data curation. Hyoung-Kyu Song: Funding acquisition; Investigation. Kihak Lee: Funding acquisition; Supervision. Hyeonjoon Moon: Supervision.

References

Akinosho, T. D., L. O. Oyedele, M. Bilal, A. O. Ajayi, M. D. Delgado, O. O. Akinade, and A. A. Ahmed. 2020. "Deep learning in the construction industry: A review of present status and future innovations." *J. Build. Eng.* 32 (Nov): 101827. https://doi.org/10.1016/j.jobe.2020.101827.

Cai, Z., S. Liu, G. Wang, Z. Ge, X. Zhang, and D. Huang. 2024. "Align-DETR: Enhancing end-to-end object detection with aligned loss." Preprint, submitted December 23, 2024. https://arxiv.org/abs/2304.07527.

Carion, N., F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. 2020. "End-to-end object detection with transformers." In *Proc., European Conf. on Computer Vision*, 213–229. Cham, Switzerland: Springer.

Dang, L. M., H. Wang, Y. Li, L. Q. Nguyen, T. N. Nguyen, H.-K. Song, and H. Moon. 2023a. "Lightweight pixel-level semantic segmentation and analysis for sewer defects using deep learning." *Constr. Build. Mater.* 371 (Mar): 130792. https://doi.org/10.1016/j.conbuildmat.2023.130792.

Dang, M., H. Wang, T.-H. Nguyen, L. Tightiz, L. D. Tien, T. N. Nguyen, and N. P. Nguyen. 2023b. "CDD-TR: Automated concrete defect investigation using an improved deformable transformers." *J. Build. Eng.* 75 (Sep): 106976. https://doi.org/10.1016/j.jobe.2023.106976.

de Brito, J., C. Pereira, J. D. Silvestre, and I. Flores-Colen. 2020. "Expert knowledge-based inspection systems." In *Inspection, diagnosis and repair of the building envelope*. Cham, Switzerland: Springer.

Dong, J., S. Roth, and B. Schiele. 2020. "Deep wiener deconvolution: Wiener meets deep learning for image deblurring." In Vol. 33 of *Proc., Advances in Neural Information Processing Systems*, 1048–1059. Red Hook, NY: Curran Associates.

Dong, X., J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo. 2022. "Cswin transformer: A general vision transformer backbone with cross-shaped windows." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 12124–12134. New York: IEEE.

Duan, K., S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. 2019. "Centernet: Keypoint triplets for object detection." In *Proc., IEEE/CVF Int. Conf. on Computer Vision*, 6569–6578. New York: IEEE.

Faqih, F., and T. Zayed. 2021. "Defect-based building condition assessment." *Build. Environ.* 191 (Mar): 107575. https://doi.org/10.1016/j.buildenv.2020.107575.

Gao, X., M. Jian, M. Hu, M. Tanniru, and S. Li. 2019. "Faster multidefect detection system in shield tunnel using combination of FCN and faster RCNN." *Adv. Struct. Eng.* 22 (13): 2907–2921. https://doi.org/10.1177/1369433219849829.

Guo, F., Y. Qian, J. Liu, and H. Yu. 2023. "Pavement crack detection based on transformer network." *Autom. Constr.* 145 (Jan): 104646. https://doi.org/10.1016/j.autcon.2022.104646.

Guo, M.-H., T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu. 2022. "Attention mechanisms in computer vision: A survey." *Comput. Visual Media* 8 (3): 331–368. https://doi.org/10.1007/s41095-022-0271-y.

Han, K., Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, and Y. Xu. 2023. "A survey on vision transformer." *IEEE Trans. Pattern Anal. Mach. Intell.* 45 (1): 87–110. https://doi.org/10.1109/TPAMI.2022.3152247.

Huang, X., Y. Li, Y. Bao, and X. Zhu. 2024a. "Sparse cross-transformer network for surface defect detection." *Sci. Rep.* 14 (1): 24731. https://doi.org/10.1038/s41598-024-75680-y.

Huang, Y.-X., H.-I. Liu, H.-H. Shuai, and W.-H. Cheng. 2024b. "DQ-DETR: DETR with dynamic query for tiny object detection." In *Proc., European Conf. on Computer Vision*, 290–305. Cham, Switzerland: Springer.

Kim, E., H. Ji, J. Kim, and E. Park. 2022. "Classifying apartment defect repair tasks in South Korea: A machine learning approach." *J. Asian Archit. Build. Eng.* 21 (6): 2503–2510. https://doi.org/10.1080/13467581.2021.1972808.

Korea. 2008. "Enforcement decree of the building act." Accessed May 11, 2023. https://elaw.klri.re.kr/eng_mobile/viewer.do?hseq=51319&type=sogan&key=4.

Lee, K., G. Hong, L. Sael, S. Lee, and H. Y. Kim. 2020. "Multidefectnet: Multi-class defect detection of building façade based on deep convolutional neural network." *Sustainability* 12 (22): 9785. https://doi.org/10.3390/su12229785.

Li, C., W. Pan, R. Su, and P. Yuen. 2022a. "Multiple structural defect detection for reinforced concrete buildings using YOLOV5s." *HKIE Trans.* 29 (2): 141–150. https://doi.org/10.33430/V29N2THIE-2021-0033.

Li, F., H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang. 2022b. "DN-DETR: Accelerate DETR training by introducing query denoising." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 13619–13627. New York: IEEE.

Li, Y., H. Wang, L. M. Dang, H.-K. Song, and H. Moon. 2022c. "Vision-based defect inspection and condition assessment for sewer pipes: A comprehensive survey." *Sensors* 22 (7): 2722. https://doi.org/10.3390/s22072722.

Lin, T.-Y., P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. 2017a. "Feature pyramid networks for object detection." In *Proc., IEEE Conf. on Computer Vision and Pattern Recognition*, 2117–2125. New York: IEEE.

Lin, T.-Y., P. Goyal, R. Girshick, K. He, and P. Dollár. 2017b. "Focal loss for dense object detection." In *Proc., IEEE Int. Conf. on Computer Vision*, 2980–2988. New York: IEEE.

Liu, S., F. Li, H. Zhang, X. Yang, X. Qi, H. Su, J. Zhu, and L. Zhang. 2022. "DAB-DETR: Dynamic anchor boxes are better queries for DETR." Preprint, submitted March 30, 2022. https://arxiv.org/abs/2201.12329.

Liu, S., T. Ren, J. Chen, Z. Zeng, H. Zhang, F. Li, H. Li, J. Huang, H. Su, and J. Zhu. 2023. "Detection transformer with stable matching." In *Proc., IEEE/CVF Int. Conf. on Computer Vision*, 6491–6500. New York: IEEE.

Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. 2016. "SSD: Single shot multibox detector." In *Proc., 14th European Conf. on Computer Vision*, 21–37. Cham, Switzerland: Springer.

Mundt, M., S. Majumder, S. Murali, P. Panetsos, and V. Ramesh. 2019. "Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 11196–11205. New York: IEEE.

Nam, H., G. Kwon, G. Y. Park, and J. C. Ye. 2024. "Contrastive denoising score for text-guided latent diffusion image editing." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 9192–9201. New York: IEEE.

Parmar, N., A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. 2018. "Image transformer." In *Proc., Int. Conf. on Machine Learning*, 4055–4064. Stockholm, Sweden: Proceedings of Machine Learning Research.

Perez, H., J. H. Tah, and A. Mosavi. 2019. "Deep learning for detecting building defects using convolutional neural networks." *Sensors* 19 (16): 3556. https://doi.org/10.3390/s19163556.

Ren, S., K. He, R. Girshick, and J. Sun. 2017. "Faster R-CNN: Towards real-time object detection with region proposal networks." *IEEE Trans. Pattern Anal. Mach. Intell.* 39 (6): 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.

Shi, P., H. Chen, Z. Geng, X. Fan, and Y. Xin. 2025. "Crack-ConvT Net: A convolutional transformer network for crack segmentation in underwater dams." *Complex Intell. Syst.* 11 (8): 358. https://doi.org/10.1007/s40747-025-01957-y.

Taye, M. M. 2023. "Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions." *Computers* 12 (5): 91. https://doi.org/10.3390/computers12050091.

Tian, Z., C. Shen, H. Chen, and T. He. 2020. "FCOS: A simple and strong anchor-free object detector." *IEEE Trans. Pattern Anal. Mach. Intell.* 44 (4): 1922–1933. https://doi.org/10.1109/TPAMI.2020.3032166.

Tulbure, A.-A., A.-A. Tulbure, and E.-H. Dulf. 2022. "A review on modern defect detection models using DCNNs–Deep convolutional neural networks." *J. Adv. Res.* 35 (Jan): 33–48. https://doi.org/10.1016/j.jare.2021.03.015.

Valero, E., A. Forster, F. Bosché, E. Hyslop, L. Wilson, and A. Turmel. 2019. "Automated defect detection and classification in ashlar masonry walls using machine learning." *Autom. Constr.* 106 (Oct): 102846. https://doi.org/10.1016/j.autcon.2019.102846.

Wang, H., Y. Li, L. M. Dang, S. Lee, and H. Moon. 2021. "Pixel-level tunnel crack segmentation using a weakly supervised annotation approach." *Comput. Ind.* 133 (Dec): 103545. https://doi.org/10.1016/j.compind.2021.103545.

Wang, N., X. Zhao, P. Zhao, Y. Zhang, Z. Zou, and J. Ou. 2019. "Automatic damage detection of historic masonry buildings based on mobile deep learning." *Autom. Constr.* 103 (Jul): 53–66. https://doi.org/10.1016/j.autcon.2019.03.003.

Wang, X., H. Gao, Z. Jia, and Z. Li. 2023. "BL-YOLOv8: An improved road defect detection model based on YOLOv8." *Sensors* 23 (20): 8361. https://doi.org/10.3390/s23208361.

Wang, X., T. Kong, C. Shen, Y. Jiang, and L. Li. 2020. "Solo: Segmenting objects by locations." In *Proc., 16th European Conf. on Computer Vision*, 649–665. Cham, Switzerland: Springer.

Wu, W., J. Weng, P. Zhang, X. Wang, W. Yang, and J. Jiang. 2022. "URetinex-Net: Retinex-based deep unfolding network for low-light image enhancement." *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 5901–5910. New York: IEEE.

Wu, Z., C. Shen, and A. Van Den Hengel. 2019. "Wider or deeper: Revisiting the ResNet model for visual recognition." *Pattern Recognit.* 90 (Jun): 119–133. https://doi.org/10.1016/j.patcog.2019.01.006.

Xiao, Y., Z. Tian, J. Yu, Y. Zhang, S. Liu, S. Du, and X. Lan. 2020. "A review of object detection based on deep learning." *Multimedia Tools Appl.* 79 (33): 23729–23791. https://doi.org/10.1007/s11042-020-08976-6.

Xu, S., M. Zhang, W. Song, H. Mei, Q. He, and A. Liotta. 2023. "A systematic review and analysis of deep learning-based underwater object detection." *Neurocomputing* 527 (Mar): 204–232. https://doi.org/10.1016/j.neucom.2023.01.056.

Zhang, G., Z. Luo, Y. Yu, K. Cui, and S. Lu. 2022a. "Accelerating DETR convergence via semantic-aligned matching." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 949–958. New York: IEEE.

Zhang, H., F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum. 2022b. "DINO: DETR with improved denoising anchor boxes for end-to-end object detection." Preprint, submitted July 11, 2022. https://arxiv.org/abs/2203.03605.

Zhu, X., W. Su, L. Lu, B. Li, X. Wang, and J. Dai. 2020. "Deformable DETR: Deformable transformers for end-to-end object detection." Preprint, submitted March 18, 2021. https://arxiv.org/abs/2010.04159.