SINERGI Vol. 30, No. 3, October 2026: 981-994
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2026.3.022

Surface crack edge-aware segmentation using hierarchical transformer encoder and Sobel edges for aerial inspection

Earl Ryan Aleluya[1,3,*], Kathleen Rose Ocaña[1], Preus Prixor Manulat[1], Francis Jann Alagon[1,3], Carl John Salaan[2,3]
[1]Department of Computer Engineering and Mechatronics, College of Engineering, Mindanao State University – Iligan Institute of Technology, Philippines
[2]Department of Mechanical Engineering and Technology, College of Engineering, Mindanao State University - Iligan Institute of Technology, Philippines
[3]Center for Mechatronics and Robotics, Mindanao State University - Iligan Institute of Technology, Philippines

Abstract
Structural crack detection remains a key component of structural health monitoring for large-scale civil infrastructure, where manual inspection is costly and hazardous. While aerial imaging enables rapid data acquisition, reliable crack analysis is limited by the difficulty of obtaining high-resolution images at close range without risking UAV damage. To address this, we introduce a UAV equipped with a protective spherical shell that allows safe surface contact. Although CNN-based architectures such as FPN and U-Net are established for crack segmentation, their encoders struggle to model long-range dependencies and fine edge structures, which result in poor delineation of thin, low-contrast, and discontinuous cracks. This limitation remains a persistent research gap in aerial crack analysis. To address this gap, this study investigates two complementary strategies: (i) replacing conventional CNN encoders with hierarchical Transformer backbones to capture global contextual relationships, and (ii) introducing edge-aware input augmentation using luminosity-based grayscale and Sobel gradient maps to optimize boundary sensitivity. We conducted experiments on our CrackUAS dataset, which consists of aerial images collected from infrastructure inspection scenarios. Integrating a hierarchical Transformer encoder with Sobel edge augmentation yields consistent performance gains across architectures. The ablation study concludes that F1-scores increase by approximately 33% for U-Net and 13% for FPN relative to their baseline configurations. These results show that integrating Transformer-based feature encoding with edge-aware input augmentation improves aerial crack segmentation.

Keywords:
Crack Detection;
Deep Learning;
Semantic Segmentation;
Structural Health Monitoring;
Unmanned Aerial Vehicles;

Article History:
Received: November 17, 2025
Revised: April 24, 2026
Accepted: April 27, 2026
Published: October 2, 2026

Corresponding Author:
Earl Ryan Aleluya
Department of Computer Engineering and Mechatronics, Mindanao State University – Iligan Institute of Technology, Philippines
Email:
earlryan.aleluya@g.msuiit.edu.ph

INTRODUCTION

Structural Health Monitoring (SHM) aims to assess the condition of infrastructure, including its materials, components, and overall assembly [1][2]. By diagnosing an infrastructure's current state, SHM prevents potential catastrophic failures. A recent review in [3] reported several established sensing technologies - temperature sensing, vibration response, local strain measurement, electrical/electrochemical sensing, ultrasonic testing, acoustic response, and automated visual inspection. Traditional SHM methods that rely on manual measurement are often labor-intensive, time-consuming, and risky [4][5]. This is why unmanned aerial vehicles (UAVs) have gained attention, as they enable rapid inspection [6].

Automated aerial visual inspection combines UAV-mounted cameras and computer

vision techniques to increase inspection throughput [7]. However, high-quality data acquisition remains challenging with off-the-shelf UAVs, as the close-proximity flight required for high-resolution imaging poses a significant risk of propeller damage if the UAV contacts infrastructure surfaces. In our previous work [8], we introduced the CrackUAS dataset and compared segmentation models, including the Feature Pyramid Network (FPN), LinkNet, U-Net, and U-Net++, all employing VGG16 pre-trained encoder weights. Among these, FPN is the fastest model with moderate prediction accuracy. On the other hand, the U-Net and U-Net++ achieved the highest F1-scores—67% and 66%, respectively—at an Intersection over Union (IoU) threshold of 0.75. However, these scores remain below the performance expected in industry applications, which represent a bottleneck in the current inspection framework.

Despite recent advances in Transformer-based vision models [9][10] and edge-aware learning [11]-[13], baseline Vision Transformers inherently suffer from a low-frequency bias. They are less sensitive to the high-frequency topological details required for aerial crack segmentation. While existing studies frequently replace CNN backbones with Transformers to capture global context, this patch-based global attention often dilutes the sharp, fragmented boundaries of fine cracks. In contrast, explicit edge-aware modifications (e.g., Sobel filtering) capture fine details but are highly susceptible to global background noise when used with traditional local convolutions.

Constraining global self-attention with spatial priors to overcome Transformers' intrinsic boundary blurring remains a critical open gap. To address this, the present study does not merely adopt an alternative backbone; rather, we propose a structurally guided representation learning approach. By explicitly injecting an edge prior (Sobel map) and luminosity channels into the input space of a hierarchical Transformer encoder, we force the multi-head self-attention mechanism to prioritize topological continuity over background textural noise. Figure 1 illustrates this early-fusion integration within the U-Net and FPN architectures, where gradients from the segmentation loss are backpropagated end-to-end. This coupling creates a synergistic effect: the edge prior anchors the Transformer's attention to fine crack boundaries. At the same time, the Transformer's global receptive field connects fragmented edge features over long distances, which suppresses the localized noise typical of raw edge maps.

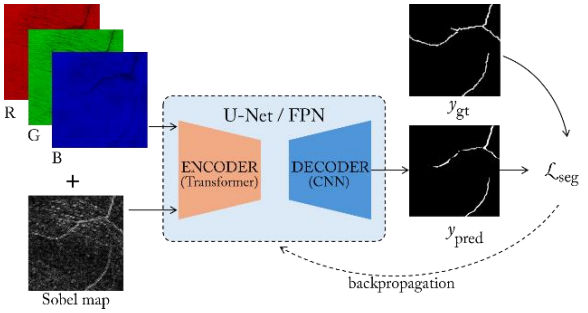

Figure 1. Overview of the improved surface crack segmentation framework showing the integration of the Sobel edge map as an additional input channel and the Transformer-based encoder within the U-Net and FPN architectures.

While recent foundation-model approaches (e.g. Segment Anything Model [14]-[17]) have demonstrated state-of-the-art generalizability, their application to spherical shell-UAV inspection is limited by two factors: (i) computational overhead: the high parameter count precludes real-time deployment; and (ii) foreground sensitivity: foundation models lack the spatial priors necessary to distinguish between the drone's protective shell and actual surface defects. Our approach maintains a lean parameter count while utilizing a Sobel-augmented attention mechanism to filter out hardware-induced artifacts. Beyond the mechanical design of a collision-resilient spherical shell, this work addresses the unique computer vision challenges inherent to aerial inspections using a spherical shell-UAV. Thus, the paper provides the following methodological contributions:

1. We introduce an obstruction-aware, structurally guided architecture specifically designed to resolve the visual interference and low-frequency bias common in close-proximity spherical UAV imagery by integrating a hierarchical Transformer encoder with U-Net and FPN decoders.
2. Rather than treating edge detection as a post-processing step, we design an augmented input space (luminosity or Sobel edge) that explicitly constrains the Transformer's global attention to high-frequency crack topologies.
3. We benchmarked our modified networks on both CrackUAS [8] and a public dataset [14], comparing them against predecessor configurations and contemporary models such as CrackW-net [15], EdgeSAM [16], CrackSAM [17], and Crack-EdgeSAM [14].

The rest of the paper is organized as follows: The related work section synthesizes relevant studies about surface crack segmentation methods and related imaging techniques.

The Methodology section discusses the materials and methods, while the following sections present the optimization search for a high-performing encoder backbone and the results of the ablation study. Lastly, the conclusion section summarizes the paper.

RELATED WORK

Improving semantic segmentation of surface cracks requires synthesizing studies on aerial inspection methods. This section reviews encoder designs and input augmentations to synthesize recent trends in solutions.

Aerial Surface Crack Inspection

Manual crack identification relies on human inspectors, which poses safety risks, wastes time, and limits accessibility [4, 18, 19]. While integrating unmanned aerial vehicles (UAVs) offers a safer, high-throughput alternative for remote infrastructure inspection [6], off-the-shelf UAVs struggle to navigate complex structural environments like bridge trusses safely [6]. To reduce collision risk, enclosing UAVs in protective spherical shells is a practical physical solution [20][21]. However, introducing this physical protection inadvertently creates a visual challenge: foreground structural occlusion, which impacts visual analysis.

To replace risky manual inspections, deep learning for semantic segmentation has emerged as a methodology for automated crack analysis [8, 22, 23]. Early benchmarks, such as the study in [24], established U-Net with a SENet backbone as a high-performing baseline, while others proposed various CNN modifications [5, 15, 25]. Recently, the paradigm has shifted toward large vision foundation models. For instance, CrackSAM [17] applies parameter-efficient fine-tuning to the segment anything model (SAM), and Crack-EdgeSAM [14] adapts EdgeSAM [16] for climbing robots, which obtains high zero-shot accuracy [22]. Despite their impressive generalization capabilities, these foundation models present a critical limitation [26]. They are highly vulnerable to visual interference from the immediate foreground. Because they are generalized to segment protuberant structures, they often misclassify or fracture crack predictions when viewing surfaces through the bars of a protective UAV shell. As a result, this remains a practical deployment constraint largely unaddressed in the current literature.

Encoder-Decoder Segmentation Architectures

Semantic segmentation evolved from symmetric encoder-decoder frameworks designed to balance contextual understanding with spatial recovery. U-Net [27] established the standard skip-connection pathway, while the feature pyramid network (FPN) [28] and LinkNet [8] introduced multi-scale pyramids and parameter efficiency, respectively.

The success of these frameworks depends on the encoder. Traditional CNN backbones like ResNet, ResNeXt [29], and Res2Net [30] revolutionized multi-scale representation. Meanwhile, the push for hardware-aware efficiency has led to models like EfficientNet [31], MobileNetV3 [32], RegNet [32], and hybrid approaches like SegFormer [10], which replace convolutional encoders with hierarchical transformers. However, these architectures share a weakness in spherical shell-caged UAV inspections. While transformers like SegFormer excel at capturing global context, their native self-attention mechanisms do not inherently distinguish between target features (cracks) and structured foreground noise (protective shells). Similarly, traditional CNNs treat these occlusions as structural texture because of their fixed receptive fields, creating a severe trade-off between computational efficiency and preserving fine-grained spatial details under obstruction.

Input Augmentation in Segmentation

To improve segmentation performance, several related works utilize input augmentation via extra data channels. For example, edge feature fusion using the Sobel operator has been employed to add detail to COVID-19 CT images [11] and concrete cracks [12]. Similarly, spectral indices [33][34] and Gabor kernels [13][35] have been used to improve contour localization and feature extraction. Recently, foundations like SAM have been used to augment inputs with generated masks and stability scores [36].

While these augmentation strategies improve boundary delineation, they possess a fundamental limitation: they treat augmented inputs strictly as additive features for enhancement. In environments with heavy structural occlusions, such as drone shells [8], amplifying edge features optimizes the obstructive noise of the shell itself. Research has not explored the inversion of this paradigm. In particular, the strategy is to use edge gradients not as additive features, but as suppressive constraints to filter out foreground obstructions.

Summary of Research Gaps

Recent innovations in aerial infrastructure inspection have transitioned from manual surveys to UAV platforms [4, 18, 19, 20, 21], and from traditional image processing to sophisticated deep

learning methodologies like SegFormer [10] and CrackSAM [17].

Despite these advances, the literature shows a significant disconnect between the physical need for UAV protective shells and the visual capabilities of current segmentation models. Existing encoder-decoder frameworks lack the native ability to ignore structured foreground noise. While input augmentations (e.g., Sobel operators [11][12] or spectral indices [33]) are substantiated, they are utilized solely to amplify features, which exacerbate visual interference when a protective shell is present.

To bridge this gap, this study introduces an obstruction-aware architecture for UAV-based crack segmentation, explicitly designed to resolve visual interference from protective spherical shells. Unlike standard edge-aware models that use Sobel gradients as auxiliary inputs, our framework pioneers their use as dynamic spatial constraints to recalibrate transformer self-attention. This targeted suppression of structured noise advances hardware-aware vision by directly addressing the physical occlusion challenges overlooked by current models.

METHOD

Problem Formulation

In this study, the problem of semantic segmentation for UAV-acquired surface crack images is formulated as a dense pixel-wise classification task, wherein each pixel of an input image $I \in \mathbb{R}^{H\times W\times C}$, with height H, width W, and channel dimension C, is assigned a label $y_{i,j} \in \{0,1\}$, indicating crack or non-crack regions. The segmentation model f_θ: $\mathbb{R}^{H\times W\times C} \rightarrow [0,1]^{H\times W}$, parameterized by learnable weights θ, learns to approximate the mapping between the input image space and the output probability mask. The objective is to minimize the segmentation loss function $\mathcal{L}_{seg} = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} l\left(f_\theta(I)_{i,j}, y_{i,j}\right)$, where $l(\cdot)$ denotes a pixel-wise classification loss such as binary cross-entropy or Dice loss. To address the memory and input-size constraints inherent in most deep segmentation architectures, *we adopt a patch-wise prediction approach*, where the original image I is partitioned into n overlapping or non-overlapping patches $\{P_k\}_{k=1}^{n}$, each of size $h \times w$ such that $h \leq H$ and $w \leq W$. The segmentation model is then applied independently on each patch $f_\theta(P_k)$, and the resulting patch-wise predictions are concatenated and spatially merged through a reconstruction function $\Psi(\cdot)$ to generate the final segmentation map $\hat{Y} = \Psi(\{f_\theta(P_k)\}_{k=1}^{n})$.

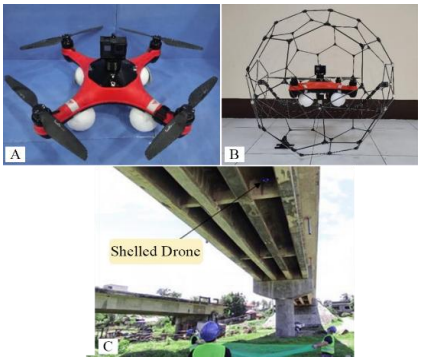

Figure 2. Illustration of the sensing platforms: (a) the base drone unit, (b) the shelled-UAV configuration, and (c) the shelled-UAV during actual deployment inside a bridge girder.

CrackUAS Dataset

Figure 2 shows the SwellPro Fisherman FD1 base drone alongside our custom shelled-UAV variant, which is waterproof and specifically designed for close-proximity inspection. While most crack detection algorithms depend on public datasets containing unobstructed surface crack images, a domain gap exists between these standard datasets and real-world UAV inspections. To bridge this gap, we constructed the CrackUAS dataset in a previous study [6] using strict operational protocols: we maintained a standard drone-to-wall distance of 1 meter. We positioned the camera perpendicular to the wall to prevent homographic errors.

Data gathering was divided into three distinct setups. The first setup utilized the shelled UAV to capture images directly from real-world infrastructure, such as building, bridge, and dam walls. The second setup used an off-the-shelf commercial drone to safely gather unobstructed crack images in controlled, less danger-prone areas. The third setup took place in a controlled laboratory environment, where the shelled UAV hovered in front of a green screen to capture images of its rotating shell.

We combined data from the second and third setups to augment the dataset with synthetic images. By removing the green background with editing software, we obtained transparent shell images, which we then superimposed onto unobstructed surface crack photographs. The CrackUAS dataset comprises three image types: (i) unobstructed crack images, (ii) real-world images from the shelled UAV showing portions of the rotating shell, and (iii) synthetic images. We extracted all data from 1920×1080-pixel flight-test videos. After filtering for quality so we annotated only clearly visible cracks, we divided the images into 448×448-pixel patches using a sliding-window approach. Across all derived patches, crack pixels

account for 2% to 34% of the total pixels, with the remainder constituting the background.

During dataset construction, four civil infrastructure experts annotated pavement distresses in UAV-acquired images. We cleaned the data by filtering out blurred images and those with a structural similarity index of at least 90% compared to the initial image. To improve model generalization and avoid overfitting, we systematically applied a data augmentation pipeline to each training patch. Specifically, this pipeline combines spatial transformations including random cropping, scaling ($0.8\times$ to $1.2\times$), and geometric flipping (horizontal and vertical with a 50% probability. It also includes pixel-level perturbations such as contrast adjustments (±20%) and Gaussian blur (using kernel sizes from 3×3 to 7×7) to simulate diverse real-world environmental conditions. To ensure a fair comparison with our previous study [6], we used the same standardized sampling strategy with a 3:1:1 split across the training, validation, and testing sets, resulting in 9,070 training images, 3,024 validation images, and 3,024 testing images. We strictly maintained this fixed split rather than using k-fold cross-validation to preserve one-to-one benchmark compatibility with prior literature.

Patch-wise Training and Inference

Both training and inference stages adopt a fixed spatial resolution of 448×448 but differ in sampling and aggregation strategies. During training, each 1920×1080 frame is decomposed into patches using a stochastic sliding-window scheme, where the overlap ratio between adjacent patches is randomly varied between 10% and 70% along both spatial dimensions. This yields about 15 base patches per image (≈5 along width and ≈3 along height), augmented with additional randomly shifted crops at the same resolution to increase spatial diversity and mitigate boundary bias. Importantly, supervision is applied at the patch level; therefore, no stitching or reconstruction of the full-resolution image occurs during training. In contrast, inference applies a deterministic tiling strategy to produce complete spatial coverage and consistent evaluation. Specifically, we generate a fixed grid of 15 patches. However, because 1920 and 1080 are not exact multiples of 448, the rightmost and bottommost patches align to the image boundaries, inducing controlled overlap with preceding patches (up to ≈70% horizontally and ≈50% vertically). For example, while the top-left patch spans coordinates (0,0)–(447,447), the final patch in the first row shifts to align with the right boundary, creating overlap with its neighboring patch. A similar adjustment is applied along the vertical axis for the bottom row. The final full-resolution prediction is obtained via an overlap-aware stitching procedure, where predictions from overlapping regions are fused using feather blending. This approach performs spatially weighted averaging based on distance-to-boundary maps, assigning higher confidence to central regions of each patch and smoothly attenuating contributions near edges.

Implementation Details

For model development, we trained all neural networks across three computing environments to ensure flexibility and reproducible results. The first environment was a laptop with an Intel Core i7 processor, 16 GB RAM, and an NVIDIA GeForce RTX 2060 GPU with 1920 CUDA cores and 240 Tensor Cores. The second environment utilized a desktop system with an Intel Core i5 processor, 16 GB RAM, and an NVIDIA GeForce RTX 2070 GPU providing 2304 CUDA cores and 288 Tensor Cores. We also utilized Google Colab Pro, which offers an NVIDIA T4 Tensor Core GPU, as an additional training platform. For network training, we set the initial learning rate to 0.001, with a decay factor of 0.5 applied every 5 epochs. Optimization uses stochastic gradient descent (SGD) with a momentum coefficient of 0.9. The model is trained for 50 epochs using a batch size of 4. For evaluation and benchmarking, we consistently test all models on the first environment to maintain uniformity across experiments.

OPTIMIZATION SEARCH OF ENCODER BACKBONE FOR U-NET AND FPN

We conduct an optimization search for the encoder backbone by systematically training and assessing several modern backbone architectures, such as ResNet, ResNetX, Res2Net, EfficientNet, MobileNet, RegNet, GENet, and SegFormer, as substitutes for the standard VGG16 encoder in U-Net and FPN models. This evaluation aims to determine the encoder configuration that provides the lowest training and validation losses for surface crack segmentation.

For all experiments, the segmentation network f_θ is trained by minimizing the Binary Cross-Entropy (BCE) loss. While Dice or hybrid losses often mitigate class imbalance in pixel-wise tasks, we selected BCE to provide a standardized optimization objective across all experiments. By using a fixed BCE loss, we ensure that performance variations are attributable only to changes in encoder depth and feature extraction capabilities, rather than to the loss function itself.

Moreover, the BCE loss is aligned with the binary pixel-wise labeling scheme $y_{i,j} \in \{0,1\}$. Specifically, for a given input image I (or patch P_k), the BCE loss is defined as,

$$l_{BCE} = -[y_{i,j} \log f_\theta(I)_{i,j} + (1 - y_{i,j}) \log(1 - f_\theta(I)_{i,j}) \quad (1)$$

The BCE loss penalizes the discrepancy between the predicted crack probability $f_\theta(I)_{i,j} \in [0,1]$ and the corresponding ground-truth label. The overall segmentation objective is given by,

$$\mathcal{L}_{seg} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{i=1}^{W} l_{BCE}(f_\theta(I)_{i,j}, y_{i,j}) \quad (2)$$

Alternatively, equivalently computed in a patch-wise manner when training on image patches $\{P_k\}_{k=1}^{n}$. To address the potential class imbalance inherent in crack detection, both data augmentation and patch-based training expose the model to foreground crack signals during each epoch. The uniformly applied BCE loss isolates the influence of encoder design and enables fair comparison between U-Net and FPN frameworks.

During training, we monitor both training and validation losses to evaluate model performance and guide model selection. The best validation loss $\mathcal{L}_{\text{val}}^{(\text{best})}$ is defined as the minimum value of the validation loss across all epochs, occurring at epoch e_b, formally expressed as,

$$\mathcal{L}_{\text{val}}^{(\text{best})} = \min_{e \in \{1,\cdots,E\}} \mathcal{L}_{\text{val}}^{(e)} \quad (3)$$

$$e_b = \arg\min_{e} \mathcal{L}_{\text{val}}^{(e)} \quad (4)$$

The training loss corresponding to this epoch, denoted as $\mathcal{L}_{\text{train}}^{(\text{best})}$, represents the training loss at the epoch where the validation loss is smallest.

$$\mathcal{L}_{\text{train}}^{(\text{best})} = \mathcal{L}_{\text{train}}^{(e_b)} \quad (5)$$

Table 1 reports the best training and validation losses. Among all encoders, the Transformer-based encoder (Segformer) achieved the smallest loss gaps, as indicated by the loss difference $|\Delta\mathcal{L}^{(\text{best})}|$. Across the dataset, the best training and validation losses $\left\{\mathcal{L}_{\text{train}}^{(\text{best})} \; and \; \mathcal{L}_{\text{val}}^{(\text{best})}\right\}$ cluster for transformer-based encoders, with $\mathcal{L}_{\text{val}}^{(\text{best})} \in [0.0409, 0.0511]$ and a consistently small generalization gap $\Delta\mathcal{L}^{(\text{best})} \leq 0.018$. In contrast, CNN-based encoders show a wider dispersion in $\Delta\mathcal{L}^{(\text{best})}$, ranging from 0.0369 up to 0.1111.

After optimization, we selected the hierarchical Transformer encoder because its generalization behavior is reflected in the smaller empirical-validation loss gap, $\Delta L^{(best)}$. Furthermore, we chose SegFormer over standard Vision Transformer (ViT) architectures because of its hierarchical structure, lightweight all-MLP decoder, and avoidance of computationally expensive positional encodings.

We utilize the SegFormer-B0 variant, which comprises a lightweight profile of approximately 3.75 million parameters. The underlying hierarchical Transformer encoder is structured across four stages with block depths of [2, 2, 2, 2]. As illustrated in Figure 3 (top), the hybrid architecture couples a multi-stage Transformer encoder that maps $I \in \mathbb{R}^{H \times W \times C}$ into hierarchical feature representations $\{\mathbf{F}_s\}_{s=1}^{4}$ with a U-Net decoder, where self-attention $\sigma\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$ explicitly models long-range spatial dependencies, where $\mathbf{Q}$ is matrix representing the set of query vectors, $\mathbf{K}^T$ is the transposed matrix of key vectors, d is dimensionality of the query and key vectors, and $\mathbf{V}$ matrix of value vectors containing the actual information to be aggregated. Each query encodes the representation of a token (or position) that is seeking contextual information from other tokens. At the same time, each key serves as an index or descriptor for matching against queries. The differences between training and validation losses, as shown in Table 1, indicate that ResNet101 has the smallest loss gap, next to the Transformer.

Meanwhile, RegNetX produces the largest loss gap. Compared with all CNN-based decoders, the proposed Transformer decoder generalizes better, as reflected in a significantly smaller training–validation loss gap. Specifically, the loss difference is reduced to 0.0102, compared to 0.0369 for ResNet101 and 0.1111 for RegNetX. This reduction indicates that the Transformer-based design avoids overfitting and provides consistent feature reconstruction across training and validation distributions, likely due to the global self-attention mechanism.

The bottom illustration in Figure 3 presents the hierarchical Transformer encoder coupled with an FPN decoder, where multi-scale features $\{\mathbf{F}_s\}_{s=1}^{4}$ are projected and fused into a unified feature pyramid $\mathbf{P} = \sum_s \varphi_s(\mathbf{F}_s)$, where $\varphi_s(.)$ denotes a scale-specific feature transformation (or projection) operator applied to the feature map $\mathbf{F}_s$ before fusion. This formulation generates scale-consistent feature aggregation within the FPN. It improves generalization across varying object scales in UAV-acquired crack imagery. Quantitatively, the Transformer-based decoder shows the lowest training–validation loss gap (0.0018), compared to 0.0369 for ResNet101 and 0.1111 for RegNetX.

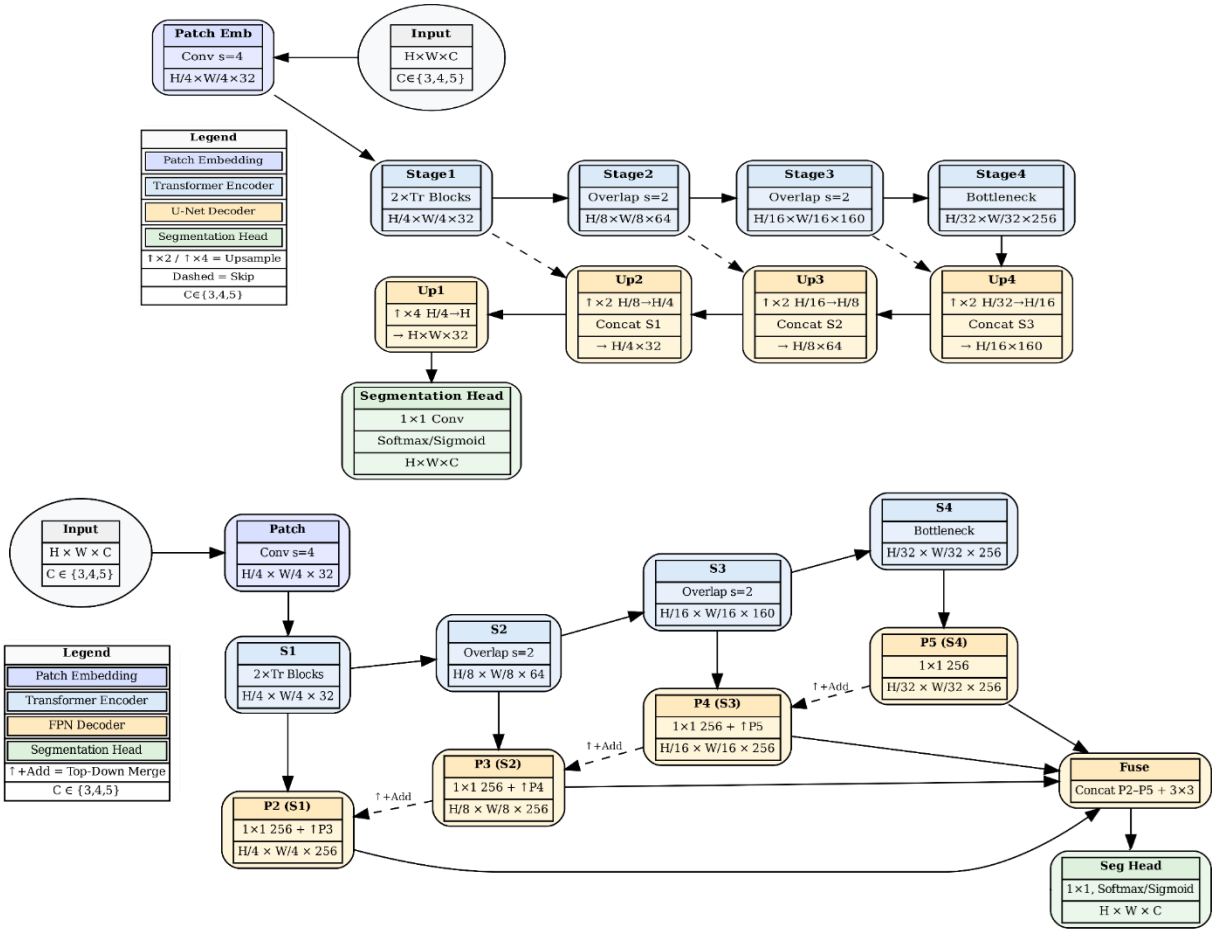

Figure 3. Overall architecture of the proposed hybrid segmentation framework: (top) Transfomer U-Net variant and (bottom) Transformer-FPN variant.

Table 1. Training Results Comparison

Model	Encoder	$\mathcal{L}_{\text{train}}^{(\text{best})}$	$\mathcal{L}_{\text{val}}^{(\text{best})}$	$\Delta\mathcal{L}^{(\text{best})}$
U-Net	resnet101	0.0998	0.1367	0.0369
U-Net	resnet152	0.0387	0.1068	0.0681
U-Net	resnext101_32x8d	0.0357	0.1132	0.0775
U-Net	es2net50_26w_4s	0.0382	0.1043	0.0661
U-Net	regnetx_032	0.0320	0.1432	0.1111
U-Net	regnety_064	0.0333	0.1071	0.0738
U-Net	gernet_l	0.0285	0.0956	0.0672
U-Net	efficientnet_b1	0.0335	0.0964	0.0629
U-Net	efficientnet_lite3	0.0315	0.0967	0.0652
U-Net	mobilenetv3_large	0.0305	0.1013	0.0708
U-Net	mit_b0	0.0401	0.0503	0.0102
U-Net	mit_b1	0.0391	0.0498	0.0107
U-Net	mit_b2	0.0359	0.0437	0.0078
U-Net	mit_b3	0.0341	0.0481	0.0140
U-Net	mit_b4	0.0331	0.0511	0.0180
FPN	mit_b0	0.0434	0.0452	0.0018
FPN	mit_b1	0.0432	0.0453	0.0020
FPN	mit_b2	0.0357	0.0409	0.0052
FPN	mit_b3	0.0336	0.0414	0.0078
FPN	mit_b4	0.0320	0.0410	0.0090

Note – Transformer encoders (`mit`) have smaller loss gaps than the rest

This corresponds to a 95.1% reduction relative to ResNet101 and a 98.4% reduction relative to RegNetX. Within the FPN framework, this improvement suggests that the Transformer decoder propagates multi-scale features across resolutions. Unlike CNN-based decoders, the self-attention mechanism administers global contextual integration.

ABLATION STUDY ON AUGMENTATION

After completing the optimization search, we performed ablation studies to quantify the effect of two input augmentation strategies—luminosity L and Sobel edge map S —on segmentation performance. Formally, the augmented input is expressed as $\tilde{I} = [I, L, S]$, where the Sobel operator S explicitly encodes high-frequency structural information associated with crack boundaries. Adding these additional input spaces alters the standard input layer dimensions. To accommodate this increased channel depth, we adjusted the encoder's initial patch-embedding dimensions accordingly. Consequently, the decoder architecture also required modification; we scaled the channel capacities of the MLP decoder layers to align with the adjusted embedding dimensions. The evaluation results indicate that incorporating S

consistently yields the largest performance gains for both U-Net and FPN decoders, as edge-aware features improve the precision-recall balance reflected in the F1-score. Quantitatively, Tables 2 and 3 show substantial performance gains, which can be analyzed in terms of both overall architectural improvements and the specific benefits of the augmented input.

U-Net achieved the highest system-wide leap. Transitioning from the standard VGG16 baseline (U-vgg-RGB, F1-score: 0.6789) to our proposed model with a Transformer encoder and RGB+S inputs (U-mit-RGBS, F1-score: 0.9022) attained a 32.89% relative improvement. The FPN architecture shows a similar, though less drastic, improvement, jumping from an F1-score of 0.6939 (F-vgg-RGB) to 0.7841 (F-mit-RGBS), a 12.98% relative gain.

To isolate the specific contribution of our data augmentation strategy, we compared the best models against their respective unaugmented Transformer baselines. For U-Net, adding RGB+S inputs improved the absolute F1-score from 0.8252 (U-mit-RGB) to 0.9022 (U-mit-RGBS), an isolated relative improvement of 9.33%. Similarly, for the FPN, the Augmented inputs increased the F1-score from 0.6958 (F-mit-RGB) to 0.7841 (F-mit-RGBS), a 12.69% relative improvement.

To explicitly determine the individual performance contributions of the network components, we can draw an ablation analysis from the results in Tables 2 and 3. Starting from the standard CNN baseline (U-vgg-RGB, IoU=0.5139), replacing the encoder with a hierarchical Transformer (U-mit-RGB) yields the largest isolated improvement, increasing IoU by an absolute margin of +0.1885.

Table 2. Evaluation Results Comparison of U-Net With and Without Augmented Inputs

Architecture	Encoder	IoU	F1-score
U-vgg-RGB	vgg16	0.5139	0.6789
U-mit-RGB	mit_b0	0.7024	0.8252
U-mit-RGBL	mit_b0	0.7616	0.8647
U-mit-RGBS	mit_b0	0.8218	0.9022
U-mit-RGBLS	mit_b0	0.7852	0.8797

Note – U-mit-RGBS attains 0.9022 F1-score, a 32.89% improvement over the VGG16 baseline (0.6789).

Table 3. Evaluation Results Comparison of FPN With and Without Augmented Inputs

Architecture	Encoder	IoU	F1-score
F-vgg-RGB	vgg16	0.5313	0.6939
F-mit-RGB	mit_b0	0.5335	0.6958
F-mit-RGBL	mit_b0	0.6128	0.7599
F-mit-RGBS	mit_b0	0.6449	0.7841
F-mit-RGBLS	mit_b0	0.6223	0.7672

Note – F-mit-RGBS attains 0.0.7841 F1-score, a 12.98% improvement over the VGG16 baseline (0.6939).

This gain presents the benefit of self-attention mechanisms in modeling long-range crack dependencies. Next, isolating the impact of the proposed edge enhancement, adding the Sobel operator to the Transformer baseline (U-mit-RGBS) yields a further absolute IoU increase of +0.1194 (reaching 0.8218). This step-by-step summary shows that while the Transformer architecture establishes an optimized baseline, the Sobel-augmented input improves spatial boundaries and maximizes overall segmentation accuracy.

Interestingly, including luminosity information L, either alone or jointly with Sobel features ($\tilde{I} = [I, L, S]$), does not yield further performance gains, which can be theoretically attributed to feature redundancy in the learned representation space. Once edge information $S \propto \|\nabla I\|$ is provided, the conditional mutual information $\mathcal{I}(y; L \mid S, I)$ becomes marginal. It indicates that intensity-based cues contribute little additional discriminative power beyond structural gradients for crack–background separation. Under this condition, the segmentation function f_θ primarily depends on high-frequency boundary features propagated through the encoder–decoder pathway. In particular, the U-Net decoder, with its symmetric up-sampling (via matrix $\mathbf{U}$) and skip connections (via $\widetilde{\mathbf{F}_s} = \phi([\mathbf{U}(\mathbf{F}_s + 1), \mathbf{F}_s])$), better preserves fine-scale crack geometry. In contrast, the FPN decoder's scale-aggregated representation $\mathbf{P} = \sum_s \varphi_s(\mathbf{F}_s)$ emphasizes global consistency that supports thin crack structures.

Table 4 presents the mean F1-score, standard deviation, median, 95% bootstrap confidence interval, and mean improvement (Δ) over the two VGG-based baselines across all ten model configurations evaluated on $n = 3{,}024$ samples from the test set. Wilcoxon signed-rank tests verify statistically significant differences between all pairwise comparisons ($p < 0.0001$). Among UNet-based architectures, the VGG encoder baseline (U-vgg-RGB) achieved a mean F1-score of 0.6745, while replacing the encoder with SegFormer (U-mit-RGB) obtained a gain of +0.1269. The luminosity channel (U-mit-RGBL) provided only marginal improvement (+0.1355 vs. baseline), while the proposed Sobel-augmented U-mit-RGBS achieved the highest mean F1-score (0.8794; median: 0.9003; 95% CI: [0.8755, 0.8834]), improving by +0.2049 over the UNet-VGG baseline and outperforming U-mit-RGBLS (0.8565). This indicates that Sobel edges provide the dominant complementary cue, while adding luminosity introduces redundancy that slightly reduces performance.

Table 4. Comparative F1-score Performance of Segmentation Architectures across Input Channel Configurations ($n = 3{,}024$)

Architecture	Input	Mean	Std	Median	95% CI	Δ vs U-VGG	Δ vs F-VGG	p-value*
U-vgg-RGB	RGB	0.6745	0.2211	0.6843	[0.6665, 0.6823]	—	+0.0107	<0.0001
U-mit-RGB	RGB	0.8014	0.1635	0.8214	[0.7958, 0.8070]	+0.1269	—	<0.0001
U-mit-RGBL	RGB+L	0.8100	0.2011	0.8605	[0.8028, 0.8172]	+0.1355	—	<0.0001
U-mit-RGBS	RGB+S	0.8794	0.1150	0.9003	[0.8755, 0.8834]	+0.2049	—	<0.0001
U-mit-RGBLS	RGB+L+S	0.8565	0.1399	0.8846	[0.8515, 0.8612]	+0.1820	—	<0.0001
F-vgg-RGB	RGB	0.6637	0.2920	0.7049	[0.6532, 0.6740]	+0.0107	—	<0.0001
F-mit-RGB	RGB	0.6826	0.2231	0.6939	[0.6742, 0.6909]	+0.0081	—	<0.0001
F-mit-RGBL	RGB+L	0.7246	0.2406	0.7591	[0.7164, 0.7331]	+0.0502	—	<0.0001
F-mit-RGBS	RGB+S	0.7739	0.1583	0.7847	[0.7680, 0.7798]	+0.0994	—	<0.0001
F-mit-RGBLS	RGB+L+S	0.7536	0.1754	0.7667	[0.7478, 0.7597]	+0.0791	—	<0.0001

* All Wilcoxon signed-rank tests are significant at p < 0.0001. Δ values indicate mean F1-score difference relative to the respective VGG baseline.

The FPN backbone followed a consistent but less pronounced trend; F-mit-RGBS achieved the best result among FPN variants at a mean F1-score of 0.7739 (+0.1101 over F-vgg-RGB), again surpassing both the RGB-only SegFormer variant and the RGB+L+S combination. The lower scores for FPN variants indicate that UNet's skip-connection architecture is better suited to using the additional edge-channel information.

Results from the two backbones show that the proposed RGB+S input modality consistently improves segmentation performance across both decoder architectures. Figures 4 and 5 illustrate representative inference results for U-Net and FPN, respectively.

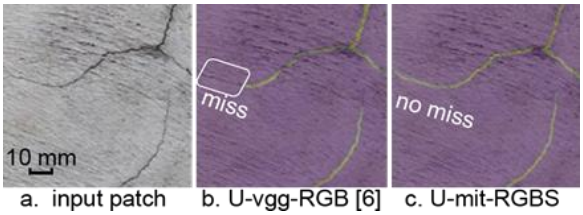

Figure 4. Representative outputs using a U-Net architecture: (a) input image (448×448), (b) prior study [6] with false negatives, and (c) our method without missed detections

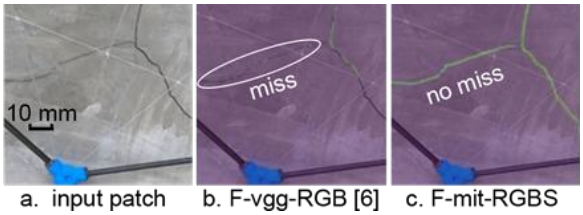

Figure 5. Qualitative comparison of segmentation performance under an FPN framework: (a) input image (448 × 448), (b) prior study [6] showing missed detections, and (c) our model achieving no omissions.

These can be interpreted through the spatial consistency of the predicted probability field $f_\theta(I) \in [0,1]^{H\times W}$. In the baseline U-Net and FPN models, as shown in the middle images, they are limited in modeling long-range dependencies, leading to fragmented predictions characterized by high local variance $\mathcal{V}(f_\theta(I)_{i,j})$ along crack trajectories. In contrast, Transformer-applied variants use global self-attention to improve spatial coherence, as shown in the rightmost images.

Despite overall improvements in comparative metrics, specific failure modes, namely, over-segmentation and occlusion-induced discontinuities, remain. These two modes persist in Figures 6 and 7, respectively.

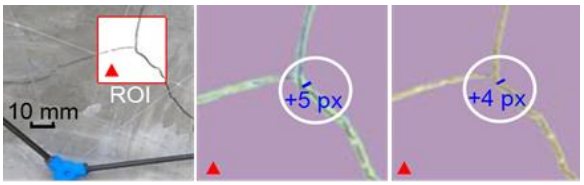

Figure 6. Over-segmentation example: (a) input image (448×448) with region of interest or ROI, (b) hybrid FPN prediction (+5 pixels excess), and (c) hybrid U-Net prediction (+4 pixels excess)

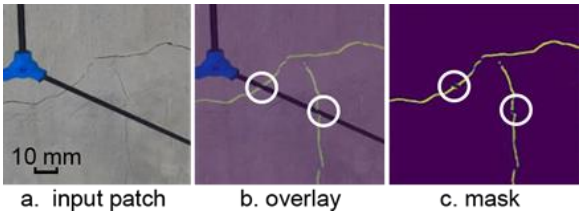

Figure 7. Shell-induced occlusion causing discontinuities in crack segmentation: (a) input patch (448 × 448), (b) U-mit-RGBS prediction overlay, and (c) mask isolating predicted cracks without shell pixels

To move beyond brief visual confirmation to quantitative analysis, we evaluated pixel-level False Positive (FP) and False Negative (FN) distributions within these problematic localized patches.

As illustrated in Figure 6, over-segmentation manifests as excessive pixel prediction beyond the true crack boundaries. This is theoretically rooted in the pixel-wise classifier's $f_\theta(I)_{i,j}$ decision boundary behavior. Edge-enhanced inputs increase gradient magnitudes $\|\nabla f_\theta(I)\|$ to capture boundaries, but this occasionally causes classifier uncertainty to expand the crack region into adjacent background areas. Visually and quantitatively, this produces a clear dilation effect in failure regions, where predicted crack trajectories are thicker than the ground truth by an average of +4 pixels for U-Net and +5 pixels for the proposed method.
For FPN. This artificial inflation directly increases localized false positives. In a patch-level analysis of Figure 6's rightmost U-Net result, the localized patch-level IoU fell to 0.75, which shows degradation compared to the 0.8218 average IoU previously reported for U-mit-RGBS in Table 2.

In contrast, Figure 7 shows failure cases where an occluding spherical shell creates definitive missing-support regions where the input image support $I_{i,j}$ is undefined. The visual evidence establishes that discontinuities in the yellow U-Net prediction line (circled) occur strictly within these occluded areas. Predicted crack lines visibly become thinner (fade out) as they approach the occluding boundary before breaking completely. Our localized patch analysis indicates that insufficient spatial contextual support leads to a marked increase in false negatives. In heavily occluded regions, such as shown in Figure 7, the patch-level IoU decreases to approximately 0.67. Compared with U-mit-RGBS in Table 2, this reduction suggests that although the proposed models generalize well, occlusion still affects prediction performance.

Regarding computational cost, the transition from the standard RGB pipeline to the augmented RGBS input pipeline introduces negligible overhead. The necessary architectural adjustments are restricted to the initial patch embedding layer to accommodate the additional input channel, while the core macro-architecture remains identical across both variants. The core macro-architecture consists of the number of Transformer blocks and self-attention heads. The primary source of parameter difference between F-mit-RGBS and U-mit-RGBS lies not in the input modification. However, the decoder choice differs: the FPN decoder adds about 0.8M parameters beyond the MiT-B0 encoder, whereas the UNet-style decoder adds about 2.5M due to heavier upsampling blocks with skip connections. Accordingly, F-mit-RGBS and U-mit-RGBS report total parameters of approximately 4.2M and 5.9M, and GFLOPs of 6.6 and 7.9 at 448×448 input resolution, respectively. Both models remain lighter than the SAM-based counterparts. Despite this compactness, U-mit-RGBS achieves an IoU of 0.553 and an F1-score of 0.712, while delivering a total throughput of around 3 FPS.

Comparison with State-of-the-Art Methods

To ensure a fair, standardized comparison with existing state-of-the-art methods, we evaluated Table 5 on a different dataset. This dataset is a public repository of 1,695 images by Khanhha et al., as used in [14]. We used this dataset rather than the custom CrackUAS dataset used in Tables 2 and 3. We performed zero-shot inference on this public dataset to benchmark our best models (U-Net and FPN with Transformer encoders and RGB+S inputs). Specifically, we evaluated the models directly without any retraining or fine-tuning on Khanhha's dataset. This cross-dataset evaluation demonstrates our models' generalization capabilities. While Khanhha's images share similarly high acquisition quality with our CrackUAS dataset, they present out-of-distribution challenges, such as lacking spherical shell obstructions and varying from our strict 1-meter wall-to-drone standoff distance. Precision (P), recall (R), and IoU values for this zero-shot evaluation are listed in Table 5, with the F1-score computed directly from the reported precision and recall.

The trends in Table 5 can be explained by the interplay among feature representation, spatial context modeling, and network architecture. Models such as CrackW-net [15] and EdgeSAM [16] struggle to balance precision and recall, leading to significant performance drops, as evidenced by their low IoU values. In contrast, architectures using hierarchical Transformers (CrackSAM [17], Crack-EdgeSAM [14], our F-mit-RGBS and U-mit-RGBS) achieve robust F1-scores and IoU. This improvement occurs because self-attention mechanisms model long-range dependencies, while skip connections preserve and propagate critical boundary cues.

To ensure a strictly fair and normalized inference-latency comparison, we did not rely on the original FPS metrics reported in [14], which were obtained using a high-end NVIDIA RTX 4070Ti. Instead, we re-evaluated the inference speed of both the baseline architectures and our proposed models on the same hardware: a single NVIDIA RTX 2060.

Table 5. Performance and Latency Comparison with State-of-the-Art Networks

Network	P	R	F1	IoU	Params (M)	GFLOPS	Inference (ms)	Overhead (ms)	Inference FPS	Total FPS
CrackW-net	0.326	0.664	0.437	0.280	31.0	182.7	303	3220.3	3.3	0.28
EdgeSAM	0.184	0.765	0.296	0.174	9.6	22.1	39.4	479	25.4	1.93
CrackSAM	0.747	0.772	0.759	0.612	91.0	458	172.4	1313.8	5.8	0.67
Crack-EdgeSAM	0.743	0.738	0.740	0.587	12.8	30.8	39.5	292.4	25.3	3.01
F-mit-RGBS	0.65	0.641	0.645	0.476	4.2	6.6	33.8	265.3	29.6	3.34
U-mit-RGBS	0.728	0.696	0.712	0.553	5.9	7.9	44.4	345.1	22.5	2.57

Note: All models were benchmarked on NVIDIA RTX 2060. GFLOPs for SAM-based models are reported at their fixed internal resolution of 1024×1024; all other models use the experimental input resolution of 448×448.

Furthermore, while the original captured UAV frames have a native resolution of 1920×1080 pixels, we resize all images to a standardized 448×448 pixels before inference. This preprocessing step reduces the computational bottleneck, keeping network latency low enough to support real-time operation. Table 5 reports the results of these standardized latency tests, categorized as inference time and system overhead. The inference time represents the computational cost of the model's forward pass. System overhead accounts for end-to-end operational costs, including image loading, preprocessing/transformation, and model initialization or unloading in memory.

Analyzing the specific precision-recall trade-offs provides insight into operational viability. While the heavy SAM-based architectures (CrackSAM, Crack-EdgeSAM) achieve slightly higher absolute metrics on this dataset, our U-mit-RGBS model maintains a highly practical, tightly balanced trade-off (Precision: 0.728, Recall: 0.696) at a significantly lower computational cost (29.6 FPS vs. CrackSAM's 5.8 FPS). In the context of UAV-based inspection, this precision-recall balance matters: the model remains highly sensitive to essential crack pixels (minimizing dangerous false negatives) while penalizing background noise and shadows to prevent excessive false alarms (minimizing false positives). Our U-mit-RGBS delivers an equal trade-off between real-time inference speed and crack segmentation.

Regarding operational robustness, our evaluation inherently reflects resilience to UAV-relevant environmental variations. First, the inherent diversity of the CrackUAS dataset addresses illumination variability, reinforced by photometric data augmentation during training. Thus, the performance on the unseen CrackUAS test set (Table 2) signifies baseline robustness to illumination changes. Second, we validate robustness to surface texture variation via cross-dataset evaluation (Table 5), where we perform zero-shot inference on Khanhha et al.'s dataset [14], which differs in concrete textures, weathering patterns, and acquisition setup. The results imply that the proposed models, particularly with RGB+S inputs, generalize beyond dataset-specific textural statistics.

Analysis of Patch Reconstruction Artifacts

The inference pipeline operates on high-resolution 1920×1080 frames by partitioning them into 448×448 patches, which introduces a nontrivial risk of patch reconstruction artifacts. In conventional non-overlapping tiling schemes, rigid patch boundaries produce sharp, high-frequency discontinuities along seam locations. In aerial surface crack detection, such artifacts are particularly problematic because these artificial linear structures can resemble genuine structural cracks, potentially increasing false-positive rates along patch borders.

To address this issue, we used an overlap-aware deterministic tiling strategy in which the rightmost and bottommost patches are boundary-aligned, inducing controlled overlap of up to approximately 70% horizontally and 50% vertically. Overlapping prediction regions are then reconciled via distance-to-boundary feather blending, which performs spatially weighted averaging that assigns higher confidence to central patch regions and smoothly attenuates contributions near edges.

To quantify the impact of this strategy, Table 6 reports segmentation performance, measured by IoU, for configurations with and without feather blending. While predictions are generated for all patches derived from the 1920×1080 frames, the evaluation specifically focuses on edge-adjacent patches, namely those along the rightmost and bottommost regions. These portions are where stitching artifacts are most likely to occur and where feather blending is applied.

Table 6. Effect of Feather Blending on Patch-Boundary IoU

Architecture	Blending	Mean	Std	ΔIoU
F-mit -RGBS	×	0.631	0.172	—
F-mit-RGBS	✓	0.645	0.159	+0.014
U-mit-RGBS	×	0.805	0.138	—
U-mit-RGBS	✓	0.822	0.126	+0.017

We report results for both proposed architectures: U-mit-RGBS and F-mit-RGBS. As shown in Table 6, feather blending provides a marginal yet consistent improvement in IoU across both architectures. For U-mit-RGBS, the mean IoU increases from 0.8050 to 0.8218 (ΔIoU = +0.017), with the standard deviation decreasing from 0.138 to 0.126. Similarly, F-mit-RGBS shows an increase in mean IoU from 0.6310 to 0.6449 (ΔIoU = +0.014), accompanied by a reduction in standard deviation from 0.172 to 0.159. These gains, while limited in magnitude, align with the localized effect of feather blending on edge-adjacent patches.

CONCLUSION

In this study, we investigated hierarchical Transformer encoders integrated with U-Net and FPN decoders for aerial surface crack segmentation, introducing Sobel edge maps as an explicit spatial prior through input augmentation. The results show performance gains of 32.89% and 12.98% in F1-score over VGG16-based baselines for U-Net and FPN, respectively, while maintaining a lean computational footprint of 4.2M to 5.9M parameters at 22–30 FPS on a consumer-grade GPU. Zero-shot evaluation on an independent public dataset further affirms that the proposed models generalize to unseen surface textures and acquisition conditions without retraining. These findings establish that structurally guided, edge-aware Transformer segmentation is a generalizable framework for UAV-based crack inspection, particularly in spherical shell-protected deployment scenarios where foundation models are precluded by foreground occlusion sensitivity and computational overhead.

Several directions remain open for future work. First, the static Sobel operator could be replaced with a lightweight learnable edge extractor trained end-to-end with the segmentation objective to better adapt to domain-specific noise. Second, a systematic evaluation of larger Transformer variants under the RGBS scheme would help characterize the accuracy–efficiency trade-off across backbone scales. Finally, while the current patch-wise inference already achieves adequate results, occasional crack discontinuities still arise, primarily under strong inter-class similarity. This issue suggests that boundary-aware stitching could further refine continuity.

ACKNOWLEDGMENT

This research was partially supported by the Department of Science and Technology-Philippine Council for Industry, Energy, and Emerging Technology Research and Development (DOST-PCIEERD).

REFERENCES

[1] G. Wang and J. Ke, “Literature Review on the Structural Health Monitoring (SHM) of Sustainable Civil Infrastructure: An Analysis of Influencing Factors in the Implementation,” *Buildings*, vol. 14, no. 2, 2024, doi: 10.3390/buildings14020402

[2] V. Plevris and G. Papazafeiropoulos, “AI in Structural Health Monitoring for Infrastructure Maintenance and Safety,” *Infrastructures*, vol. 9, no. 12, 2024, doi: 10.3390/infrastructures9120225

[3] A. Mardanshahi, A. Sreekumar, X. Yang, S. K. Barman, and D. Chronopoulos, “Sensing Techniques for Structural Health Monitoring: A State-of-the-Art Review on Performance Criteria and New-Generation Technologies,” *Sensors*, vol. 25, no. 5, 2025, doi: 10.3390/s25051424

[4] G. Scarselli and F. Nicassio, “Machine Learning for Structural Health Monitoring of Aerospace Structures: A Review,” *Sensors*, vol. 25, no. 19, 2025, doi: 10.3390/s25196136

[5] R. Alfanz, R. Fahrizal, T. P. Utomo, T. Firmansyah, F. Muhammad, and I. M. Muztahidul, “Implementation of wavelet method and backpropagation neural network on road crack detection based on image processing,” *SINERGI*, vol. 28, no. 3, p. 489–496, July 2024, doi: 10.22441/sinergi.2024.3.005

[6] S. A. H. Mohsan, M. A. Khan, F. Noor, I. Ullah, and M. H. Alsharif, “Towards the unmanned aerial vehicles (UAVs): A comprehensive review,” *Drones*, vol. 6, no. 6, p. 147, 2022, doi: 10.3390/drones6060147

[7] A. Purwanto, D. H. Budiarti, F. N. Purnamastuti, I. Y. Tanasa, Y. Guno, A. S. Yunata, M. Wibowo, A. Hidayat, and D. Dirgahayu, “Image segmentation in aerial imagery: A review,” *SINERGI*, vol. 27, no. 3, p. 343–360, Sep. 2023, doi: 10.22441/sinergi.2023.3.006

[8] M. S. Arda, E. R. M. Aleluya, C. Y. Cahig, L. G. Librado, M. U. Sumagayan, K. M. A. Aldueso, C. M. J. Galangque, and C. J. O. Salaan, “Semantic Segmentation Models for Crack Detection: Using Shelled Unmanned Aerial Vehicle Imagery,” in *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM),* IEEE, 2021,

pp. 1–6, doi: 10.1109/HNICEM54116.2021.9731866

[9] J. M. Goo, X. Milidonis, A. Artusi, J. Boehm, and C. Ciliberto, “Hybrid-Segmentor: Hybrid approach for automated fine-grained crack segmentation in civil infrastructure,” *Automation in Construction*, vol. 170, p. 105960, 2025, doi: 10.1016/j.autcon.2024.105960

[10] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, “SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers,” in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 12077–12090, doi: 10.48550/arXiv.2105.15203

[11] F. Lu, C. Tang, T. Liu, and Z. Zhang, “SMA-Net: Sobel Operator Combined with Multi-attention Networks for COVID-19 Lesion Segmentation,” in *Digital Multimedia Communications*. Singapore: Springer Nature Singapore, 2023, pp. 377–390, doi: 10.1007/978-981-99-0856-1_28

[12] G. Liu, W. Ding, J. Shu, A. Strauss, and Y. Duan, “Two-Stream Boundary-Aware Neural Network for Concrete Crack Segmentation and Quantification,” *Structural Control and Health Monitoring*, vol. 2023, no. 1, p. 3301106, 2023, doi: 10.1155/2023/3301106

[13] L. Chen, M. Lv, J. Cai, Z. Guo, and Z. Li, “U-Net-Embedded Gabor Kernel and Coaxial Correction Methods to Dorsal Hand Vein Image Projection System,” *Applied Sciences*, vol. 13, no. 20, 2023, doi: 10.3390/app132011222

[14] Y. Wang, J. He and S. Yu, "CrackESS: A Self-Prompting Crack Segmentation System for Edge Devices," *2025 IEEE 28th International Conference on Intelligent Transportation Systems (ITSC)*, Gold Coast, Australia, 2025, pp. 1-6, doi: 10.1109/ITSC60802.2025.11423250.

[15] C. Han, T. Ma, J. Huyan, X. Huang, and Y. Zhang, “CrackW-Net: A Novel Pavement Crack Image Segmentation Convolutional Neural Network,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 22 135–22 144, 2022, doi: 10.1109/TITS.2021.3095507

[16] C. Zhou, X. Li, C. C. Loy, and B. Dai, “EdgeSAM: Prompt-In-the-Loop Distillation for SAM,” *International Journal of Computer Vision*, vol. 133, no. 12, p. 8452–8468, Sep. 2025, doi: 10.1007/s11263-025-02562-9

[17] K. Ge, C. Wang, Y. Guo, Y. Tang, Z. Hu, and H. Chen, “Fine-tuning vision foundation model for crack segmentation in civil infrastructures,” *Construction and Building Materials*, vol. 431, p. 136573, 2024, doi: 10.1016/j.conbuildmat.2024.136573

[18] M. M. O. Maluya, E. R. M. Aleluya, F. J. A. Alagon, S. E. Clar, C. J. O. Salaan, J. G. Opon, and M. F. P. Bahinting, “Rural Road Pavement Panel Detection with Unmanned Aerial Vehicles,” in *2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, 2023, pp. 1–6, doi:10.1109/HNICEM60674.2023. 10589124

[19] M. M. O. Maluya, E. R. M. Aleluya, J. G. Opon, and C. J. O. Salaan, “Lane Tracking for Autonomous Road Pavement Inspection with Unmanned Aerial Vehicles,” in *2023 6th International Conference on Applied Computational Intelligence in Information Systems (ACIIS)*, 2023, pp. 1–6, doi: 10.1109/ACIIS59385.2023.10367228

[20] M. Sumagayan, R. Mangorsi, E. R. Aleluya, C. John Salaan, and C. Premachandra, “Power Line Detection Using Unmanned Aerial Vehicle with Spherical Shell,” in *2022 2nd International Conference on Advanced Research in Computing (ICARC)*, 2022, pp. 160–164, doi: 10.1109/ICARC54489.2022.9753854

[21] M. U. Sumagayan, E. R. Aleluya, R. B. Mangorsi, F. J. Alagon, E. R. Aleluya, Y. A. B. Mangorsi, H. W. H. Premachandra, C. Premachandra, and H. Kawanaka, “Foreign Object Obstruction Evaluation for Distribution Network Inspection,” *IEEE Access*, vol. 12, pp. 172 325–172 342, 2024, doi: 10.1109/ACCESS.2024.3484156

[22] S. Egodawela, A. Khodadadian Gostar, H. A. D. S. Buddika, A. J. Dammika, N. Harischandra, S. Navaratnam, and M. Mahmoodian, “A Deep Learning Approach for Surface Crack Classification and Segmentation in Unmanned Aerial Vehicle Assisted Infrastructure Inspections,” *Sensors*, vol. 24, no. 6, 2024, doi: 10.3390/s24061936

[23] J. M. A. Bolaybolay, E. R. M. Aleluya, S. E. Clar, and C. J. O. Salaan, “Surface Cracks Length Calculation in Segmented Images using Image Processing Techniques,” in *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, 2022, pp. 1–6, doi: 10.1109/HNICEM57413.2022.10109286

[24] Z. Shi, N. Jin, D. Chen, and D. Ai, “A comparison study of semantic segmentation

networks for crack detection in construction materials," *Construction and Building Materials*, vol. 414, p. 134950, 2024, doi: 10.1016/j.conbuildmat.2024.134950

[25] B. Chen, H. Zhang, G. Wang, J. Huo, Y. Li, and L. Li, "Automatic concrete infrastructure crack semantic segmentation using deep learning," *Automation in Construction*, vol. 152, p. 104950, 2023, doi: 10.1016/j.autcon.2023.104950

[26] Q. Zaheer, S. Qiu, S. M. A. H. Shah, C. Ai, and J. Wang, "Intelligent Multitasking Framework for Boundary-Preserving Semantic Segmentation, Width Estimation, and Propagation Modeling of Concrete Cracks," *Journal of Infrastructure Systems*, vol. 31, no. 3, p. 04025009, 2025, doi: 10.1061/JITSE4.ISENG-2574

[27] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015, doi: 10.48550/arXiv.1505.04597

[28] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015*, Lecture Notes in Computer Science, vol 9351, 2015*,* doi : 10.1007/978-3-319-24574-4_28

[29] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, "Aggregated Residual Transformations for Deep Neural Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 5987-5995, doi: 10.1109/CVPR.2017.634.

[30] S. -H. Gao, M. -M. Cheng, K. Zhao, X. -Y. Zhang, M. -H. Yang and P. Torr, "Res2Net: A New Multi-Scale Backbone Architecture," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 2, pp. 652-662, 1 Feb. 2021, doi: 10.1109/TPAMI.2019.2938758

[31] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *Proc. Of Machine Learning Research*, 2020, doi: 10.48550/arXiv.1905.11946

[32] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollar, "Designing Network Design Spaces," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2020, pp. 10425-10433, doi: 10.1109/CVPR42600.2020.01044

[33] Y. Zhang, R. Yang, Q. Dai, Y. Zhao, W. Xu, J. Wang, and L. Wang, "Boosting Semantic Segmentation of Remote Sensing Images by Introducing Edge Extraction Network and Spectral Indices," *Remote Sensing*, vol. 15, no. 21, 2023, doi: 10.3390/rs15215148

[34] M.-D. Yang, H.-H. Tseng, Y.-C. Hsu, and H. P. Tsai, "Semantic Segmentation Using Deep Learning with Vegetation Indices for Rice Lodging Identification in Multi-date UAV Visible Images," *Remote Sensing*, vol. 12, no. 4, 2020, doi: 10.3390/rs12040633

[35] S. B. Nemade and S. P. Sonavane, "Semantic segmentation using GSAUNet," *ICT Express*, vol. 9, no. 1, pp. 1–7, 2023, doi: 10.1016/j.icte.2022.09.007

[36] Y. Zhang, T. Zhou, S. Wang, P. Liang, and D. Z. Chen, "Input Augmentation with SAM: Boosting Medical Image Segmentation with Segmentation Foundation Model," *Medical Image Computing and Computer Assisted Intervention – MICCAI 2023*, vol. 14393, 2023, pp. 129-139, doi: 10.1007/978-3-031-47401-9_13