CrackScopeNet: A Lightweight Neural Network for Rapid Cracks Detection on Resource-Constrained Drone Platforms

Tao Zhang , Liwei Qin , Quan Zou , Liwen Zhang , Rongyi Wang , Heng Zhang *

Posted Date: 9 July 2024

doi: 10.20944/preprints202407.0705.v1

Keywords: Computer vision; Crack detection; Drone platforms; Semantic segmentation; Lightweight neural network

Article

CrackScopeNet: A Lightweight Neural Network for Rapid Cracks Detection on Resource-Constrained Drone Platforms

Tao Zhang, Liwei Qin, Quan Zou, Liwen Zhang, Rongyi Wang and Heng Zhang *

College of Computer and Information Science College of software, Southwest University, Chongqing 400715, China
* Correspondence: dahaizhangheng@swu.edu.cn

Abstract: Detecting cracks in structural health monitoring is crucial for ensuring infrastructure safety and longevity. Using drones to obtain crack images and automate processing can improve the efficiency of crack detection. To address the challenges posed by the limited computing resources of edge devices in practical applications, we propose CrackScopeNet, a lightweight segmentation network model that simultaneously considers local and global crack features, suitable for deployment on drone platforms with limited computational power and memory. CrackScopeNet features a multi-scale branch to improve sensitivity to cracks of varying sizes without substantial computational overhead and a stripe-wise context attention mechanism to enhance the capture of long-range contextual information, mitigating the interference from complex backgrounds. Experimental results on CrackSeg9k dataset demonstrate that CrackScopeNet leads to a significant improvement in prediction performance, with the highest mean intersection over union (mIoU) scores reaching 82.12%, while maintaining a lightweight architecture with only 1.05M parameters and 1.58G floating point operations (FLOPs). Besides, CrackScopeNet excels in inference speed on edge devices without GPU because of its low FLOPs. CrackScopeNet contributes to the development of efficient and effective crack segmentation networks, suitable for practical structural health monitoring applications using drone platforms.

Keywords: computer vision; crack detection; drone platforms; semantic segmentation; lightweight neural network

1. Introduction

Cracks serve as early indicators of structural damage in buildings, bridges, and roads, making their detection vital for structural health monitoring. Analyzing the morphological characteristics, positional information, and extent of internal damage in cracks allows for accurate safety assessments of buildings and infrastructures[1,2]. Timely detection and repair of cracks not only reduce maintenance costs but also prevent further structural deterioration, ensuring safety and durability[3,4].

Traditional crack detection methods, such as visual inspections and manual evaluations, are often costly and inefficient, relying heavily on the expertise of inspectors, which can lead to subjective and inconsistent results[5]. Therefore, the development of objective and efficient automated crack detection methods have become a significant trend in this field. Various sensor-based methods for automatic or semi-automatic crack detection, including crack meters, RGB-D sensors, and laser scanners, have been proposed[6–8]. Although these sensors are accurate, they are expensive and challenging to deploy on a large scale.

Advancements in computer vision technology have popularized image-based crack detection methods due to their long-distance, non-contact, and cost-effective nature. Traditional visual detection methods, such as morphological image processing[9,10], filtering[11,12], and percolation models[13], are simple to implement and computationally light but suffer from limited generalization performance. Environmental noise, such as debris around cracks, further complicates detection in practical engineering environments.

Recently, deep learning-based semantic segmentation algorithms have significantly improved the accuracy and stability of image recognition in noisy environments. These algorithms can precisely locate and label crack pixels, providing detailed information on crack distribution, width, length, and

 doi:10.20944/preprints202407.0705.v1

shape[14]. However, general scene understanding models often fail to capture the unique features of cracks, which are typically thin, long, and irregularly shaped[15,16]. Cracks can span entire images while occupying less than 5% of the width, necessitating models that effectively capture dependencies between distant pixels. While self-attention mechanisms excel at aggregating long-distance contextual information[17–19], they come with high computational costs, limiting detection speed. Additionally, cracks exhibit uneven distribution with significant size variations, necessitating multi-scale feature extraction[20–22]. Although methods like DeepLabV3+[23] and SegNext[24] capture multi-scale information, they are computationally intensive and costly for large images.

Current computer vision-based crack detection applications have seen a significant shift towards the utilization of unmanned aerial vehicles (UAVs) due to their flexibility, cost-effectiveness, and ability to cover large and hard-to-reach areas efficiently[25]. However, the computational resources available on UAV-based platforms or others edge devices are typically limited, often operating without GPUs, making it crucial to deploy lightweight models that can perform low-latency processing and analysis[26]. Researchers have proposed lightweight networks that reduce computational costs by minimizing deep down-sampling, reducing channel numbers, and optimizing convolutional design. However, reducing subsampling stages can lead to insufficient model receptive fields for covering large objects, as seen with ENet[27]. Bilateral backbone models partially address this issue. BiSeNet[28], for instance, adds a context path with fewer channels, and HrSegNet[29] maintains high-resolution features while adjusting parameters to reduce channels. Nevertheless, these two-branch structures increase computational overhead, and reducing channels can hinder the ability of model to learn relational features effectively.

Furthermore, we analyze the challenges in designing lightweight surface crack segmentation models: 1) existing methods increase computational complexity with large kernel convolutions, multiple parallel branches, and feature pyramid structures to handle various object sizes and shapes; 2) diverse crack image scenes and complex backgrounds limit lightweight model feature extraction, making it difficult to learn effective information from limited datasets; 3) subtle differences between cracked and normal areas complicate segmentation. While adding multiple skip connections and auxiliary training branches can improve accuracy, they increase memory overhead.

To address the aforementioned challenges, we propose CrackScopeNet, a lightweight segmentation network optimized for structural surface cracks. CrackScopeNet features an optimized multi-scale branching architecture and a carefully designed strip-wise context attention (SWA) module. Figure 1 compares classical semantic segmentation networks, lightweight semantic segmentation networks, and crack-specific segmentation networks with CrackScopeNet in terms of mean intersection over union (mIoU), model floating point operations (FLOPs), and parameters on the CrackSeg9k dataset. Only models with mIoU scores above 80% and FLOPs below 100G are shown. What can be seen clearly is that CrackScopeNet outperforms all shown models with significantly fewer FLOPs and parameters. This is due to the initial design consideration of capturing both local context information around small cracks and remote context information to identify complete cracks and eliminate background interference.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 July 2024 doi:10.20944/preprints202407.0705.v1

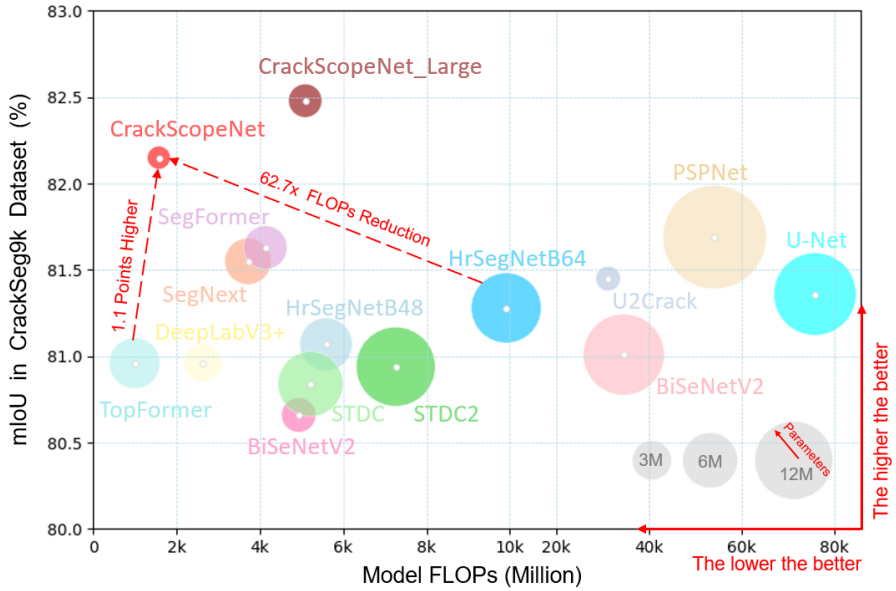

Figure 1. Comparisons between classical and lightweight semantic segmentation networks and the CrackScopeNet on CrackSeg9k dataset.

Initially, in the local feature extraction stage, we divide the channel data and perform three convolution operations with different convolution kernel sizes to obtain the local context information of cracks. Subsequently, we utilize a combination of strip pooling and one-dimensional convolution to capture remote context information without compressing channel features. Finally, we construct a lightweight multi-scale feature fusion module to aggregate shallow detail and deep semantic information. In these modules, we employ depth-separable convolution, dropout, and residual connection structures to prevent overfitting, gradient disappearance, and gradient explosion problems, resulting in a lightweight neural network adaptable to crack detection.

In summary, our main contributions are as follows.

(1) We propose a novel crack image segmentation model, called CrackScopeNet, designed to meet structural health monitoring requirements. The model effectively extracts information at multiple levels during the down-sampling stage and fuses key features during the up-sampling stage.

(2) We design a lightweight multi-scale branch module and a strip-wise context attention module to capture remote context information. These modules are meticulously crafted to align with fracture morphological characteristics, capturing rich context information while minimizing computational cost and alleviating overfitting issues common in lightweight models with fewer parameters.

(3) CrackScopeNet demonstrates state-of-the-art performance on the CrackSeg9k dataset and exhibits excellent transferability to small datasets in specific scenarios. Additionally, CrackScopeNet has a low inference delay on resource-constrained drone platforms, making it ideal for outdoor crack detection through computer vision. This ensures that drone platforms can perform rapid crack detection and analysis, enhancing the efficiency and effectiveness of structural health monitoring.

The code for this work is available on GitHub (at https://github.com/ttkingzz/CrackScopeNet) to facilitate further research and application.

2. Methods

This work introduces CrackScopeNet, a lightweight crack segmentation model designed based on crack morphological characteristics. Additionally, we propose a strip-wise attention mechanism to enhance the receptive field of convolutional neural networks, thereby improving the accuracy and continuity of crack segmentation. In the following section, we comprehensively describe our proposed method.

 doi:10.20944/preprints202407.0705.v1

2.1. Model Overview

We meticulously design feature extraction and global context attention modules based on crack morphological information, collectively named CrackScopeNet. As shown at the top of Figure 2, our proposed CrackScopeNet adopts a simple encoder-decoder structure, similar to most previous works[19,23]. It comprises a feature extractor (encoder) and a feature integrator (decoder). In the encoder, the crack image is input into the network, passing through three feature extraction stages to capture detailed and deep semantic information. Each stage comprises a down-sampling convolution layer and a series of Crack Scope (CS) Modules, each containing a multi-scale branch and an SWA module. The decoder gradually restores the crack features to the original resolution, merging different levels of feature information through the feature fusion module, thus producing an output with rich spatial and textural features. The detailed configuration of CrackScopeNet is listed in Table 1.

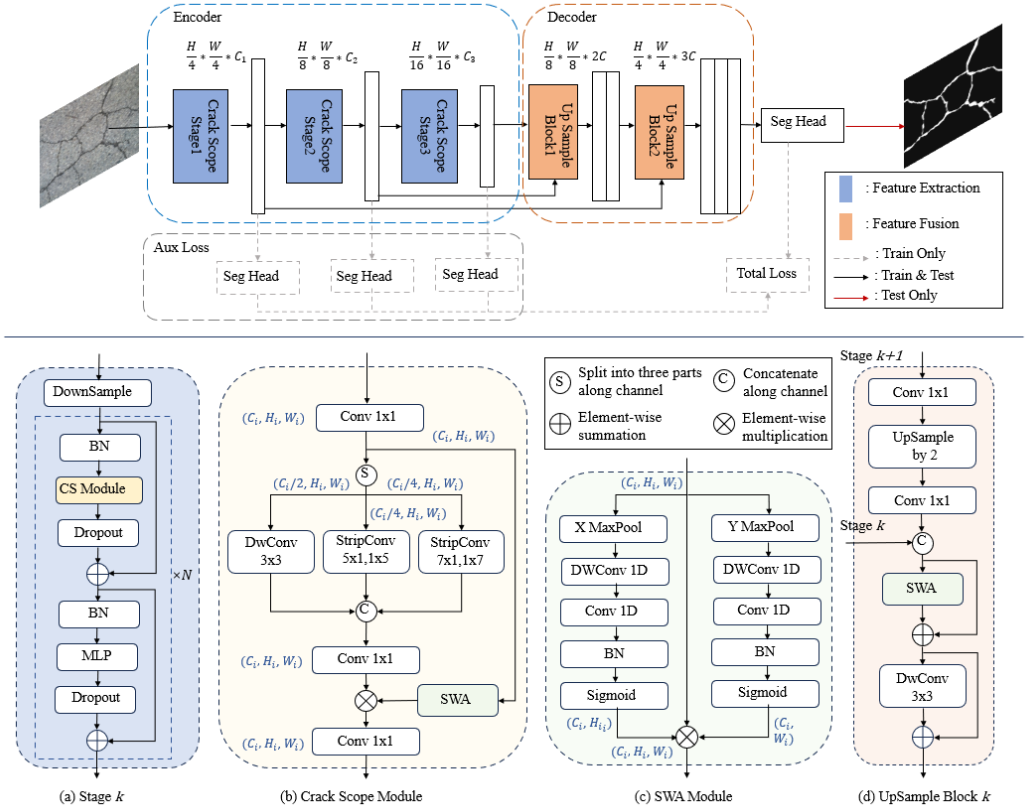

Figure 2. A structural overview of CrackScopeNet. (**a**) CrackScopeNet consists of three down-sampling stages, and each stage contains N Crack Scope Modules and FFN modules (we refer to these two combined modules as Crack Scope Block). (**b**) Crack Scope Module implies a multi-scale branch, where the input is divided into three parts along the channel dimension, and an (**c**) SWA module. (**d**) Upsampling Block upsamples deep features and stacks them with shallow information, using SWA modules and convolutional layers for feature fusion.

 doi:10.20944/preprints202407.0705.v1

Table 1. Instance of CrackScopeNet.

Stage	Downsampling			Upsampling			Output Size
	Operation	C_{in}	C_{out}	Operation	C_{in}	C_{out}	
S0	Input		3	Seg Head	96	2	400x400
S1	Stem	3	32	Concatenate	64	96	100x100
S2	CS Stage x 3	32	32	Up-samp.	128	64	100x100
S3	Donw-samp.	32	64	Concatenate	64	128	50x50
	CS Stage x 3	64	64	Up-samp.	128	64	50x50
S4	Donw-samp.	64	128				25x25
	CS Stage x 4	128	128				25x25

2.2. Feature Extraction

Inspired by SegNeXt[24], ConvNeXt[30], MetaFormer[31], and PKINet[32], we adopt a structure similar to ViT[17] for the block design at each feature extraction stage. As illustrated in Figure 2(a), each Crack Scope Block consists of two residual-connected sub-modules connected via residual connections: the Crack Scope Module and the Feed-forward Network Module. To prevent overfitting during training, we incorporate dropout layers in each sub-module. The Crack Scope Module comprises a multi-scale branch module and a Stripe-wise Context Attention (SWA) module. This section will focus on the multi-scale feature module, while the SWA module will be introduced in the next section.

Unlike general object segmentation, crack segmentation targets the identification and localization of crack areas of various shapes. Due to different development times and external influences, coarse cracks are often accompanied by fine cracks. To address the challenge of varying crack proportions in different regions, we introduce the multi-scale branch to capture texture features of different scales. As shown in Figure 2(b), the Crack Scope Module utilizes a small kernel convolution to manage local information and captures contextual information across multiple scales through parallel multi-scale branches with different channels and convolution kernel sizes.

Specifically, the Crack Scope Module employs three branches of depthwise separable convolutions to capture multi-scale local contextual information about cracks. It employs 1x1 convolutions to model the relationships between different channels. When extracting multi-scale information with a multi-branch structure, the FLOPs increase exponentially with the number of branches and the size of the convolution kernels. For regular convolution computation, given the output spatial dimensions $W * H$, input channels C_{in}, output channels C_{out}, and convolution kernel size $k_m * k_n$, the FLOPs calculation formula is expressed as follows:

$$FLOPs = W * H * C_{in} * C_{out} * k_m * k_n \tag{1}$$

To reduce computational overhead, we adjust the input and output channels and the convolution kernel sizes for the multi-scale branches. First, we divide the input features into three parts along the channels: allocating half the channels to the branch with the smallest kernel and a quarter of the channels to each of the two branches with larger kernels. After convolution computations are completed in the three branches, the features are merged along the channel dimension, and a 1x1 convolution is used to model the relationships between all channels. Recent work has shown that carefully designed large-kernel convolution networks can significantly improve model performance. For instance, ConvNeXt[30] employs 7x7 depthwise convolutions in its backbone, greatly enhancing the performance of downstream tasks. However, compared to 3x3 convolutions, 7x7 large-kernel convolutions result in more than five times computational cost. We use strip convolutions in our branch design to achieve the larger receptive field. Since cracks are mostly strip-shaped, using strip convolutions helps extract strip features. Additionally, strip convolutions are lightweight compared to standard 2D convolutions for the same receptive field[24].

Notably, considering the importance of crack detail information and computational complexity, we do not use 31x31 convolutions like RepLKNet[33] to achieve remarkable performance. Instead, we use only (1x5,5x1) strip convolutions and (1x7,7x1) strip convolutions to replace 5x5 and 7x7 2D

 doi:10.20944/preprints202407.0705.v1

convolutions for capturing local contextual information. Then, we design a remote context information attention module to assist the Crack Scope Module in obtaining global contextual information, which will be introduced below.

2.3. Stripe-wise Context Attention

The attention mechanism is an adaptive selection process that makes the network focus on important parts. As discussed above, the multi-scale branch module in the CS module is used to extract multi-scale local contextual information. We utilize strip pooling to capture long-distance contextual information inspired by GhostNetv2[34] and coordinate attention (CA)[35] in spatial dimensions. We design a Stripe-wise Context Attention (SWA) module and integrate it into the CS module. As illustrated in Figure 2(c), given the input feature x with channel number c and spatial dimensions $H * W$, we perform max-pooling operations in both height and width dimensions to obtain global width and height features Z^h and Z^w.

$$\begin{aligned} Z^h &= \max_{0 \le i < W} x(h, i) \\ Z^w &= \max_{0 \le j < H} x(j, w) \end{aligned} \tag{2}$$

Further, to avoid compressing the channel numbers like CBAM[36] and CA[35], which reduces computational load, we apply 1-dimensional depthwise separable convolutions to model the relationships across different spatial dimensions and channels. The attention representations in the horizontal and vertical directions are denoted as:

$$\begin{aligned} y^h &= \delta(F_2(F_1(F_0(Z^h)))) \\ y^w &= \delta(F_2(F_1(F_0(Z^w)))) \end{aligned} \tag{3}$$

where F_0 is a 1-dimensional depthwise separable convolution with a kernel size of 5x5, F_1 is a 1-dimensional convolution to capture inter-channel information, F_2 is batch normalization, and δ is the sigmoid activation function.

Finally, the output of the attention module can be obtained by applying the following equation:

$$Y = x * y^h * y^w \tag{4}$$

2.4. Feature Fusion

To reduce the loss of key feature information and ensure the accuracy of crack detection, we integrate the SWA module into the Up-sampling Blok for precise feature restoration and fusion. Notably, unlike most segmentation network decoders such as DeepLabV3+[23] and PSPNet[37], we do not use Atrous Spatial Pyramid Pooling (ASPP) to extract multi-scale features of high-level semantic information. On the one hand, using many dilated convolutions adds unnecessary computational overhead and increases network complexity. On the other hand, we believe that using our multi-scale branches and the SWA attention model in the encoder effectively captures both local feature information and remote dependencies of cracks, reducing the need for dilated convolutions to expand the receptive field. Further comparative details will be provided in subsequent experimental sections.

Figure 2(d) shows that high-level semantic information is adjusted through point-wise convolution operations, and bilinear interpolation restores the feature map size. It is then concatenated with features from lower stages. Subsequently, to further fuse high-level semantic features with detailed texture features, we employ an SWA module with a shortcut connection to model feature relationships across the global space and channels, fully integrating multi-scale feature maps. Next, a small kernel convolution is used to refine crack feature information. After the multi-scale fusion of the three-stage feature maps, they are fed into the segmentation head, which maps the feature map to the required segmentation output, completing the entire network computation process. Notably, to avoid the large computational overhead brought by the decoder, we do not use transposed convolutions to learn

 doi:10.20944/preprints202407.0705.v1

more parameters but only select and fuse features to construct a lightweight multi-scale feature fusion module.

2.5. Auxiliary Branch

To further enhance segmentation performance inspired by the auxiliary branches in PSPNet[37], we add three auxiliary branches in the CrackScopeNet encoder to improve the performance of feature extractor during training. The auxiliary branches are used only during training and ignored during inference, thus not affecting the inference speed of the entire network structure. However, they provide additional gradient signals, allowing higher gradient propagation to each feature extraction stage, which helps mitigate the gradient vanishing or exploding problem and improves the training stability of the encoder. For each auxiliary branch, we use the same segmentation head as the main branch and recover the original image size with different up-sampling multiples. The total loss is the weighted sum of the binary cross entropy losses from each segmentation head, as shown in the equation:

$$L_t = L_p + aL_1 + bL2 + cL3 \tag{5}$$

Here, L_t and L_p represent the total loss and primary loss, respectively. L_1, L_2 and L_3 are the losses of the three auxiliary branches. In this work, the weights a, b, and c are all set to 0.5.

3. Experimental Datasets and Setup

3.1. Datasets

In crack segmentation research, the limited size and number of publicly available datasets pose challenges for comprehensive algorithm evaluation. CrackSeg9k[38] addresses this by offering a substantial dataset specifically designed for crack segmentation tasks. Despite the CrackSeg9k images being captured with cameras and smartphones, their resolution and imaging angles are similar to UAV-captured images, allowing models to generalize well to UAV-based image data.

CrackSeg9k consists of 9,255 images with a resolution of 400 × 400 pixels, each labeled for cracks and backgrounds. As shown in Table 2, this dataset merges ten smaller sub-datasets, enhancing its diversity and robustness. It includes various crack types: linear, branched, webbed, and non-crack images, as shown in Figure 3 (a), ensuring models trained on it can generalize across different crack patterns and conditions. Notably, the creators of CrackSeg9k have also corrected label noise, boosting its reliability.

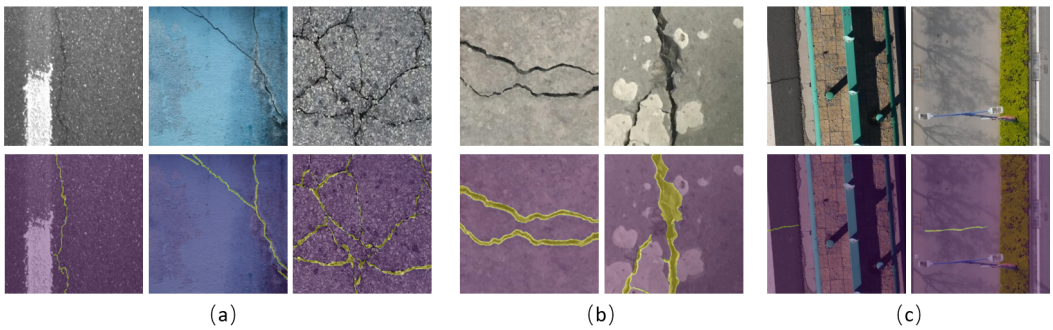

Figure 3. Samples in three crack datasets. The first line is the original images, and the second line is the overlay effect of the masks and the original images. (a) Samples in CrackSeg9k dataset. (b) Samples in Ozgenel dataset. (c) Samples in Aerial Track dataset.

 doi:10.20944/preprints202407.0705.v1

Table 2. Sub-datasets in CrackSeg9k.

Name	Number	Material
Masonry[39]	240	Masonry structures
CFD[40]	118	Paths and sidewalks
CrackTree200[41]	175	Pavement
Volker[42]	427	Concrete structures
DeepCrack[43]	443	Concrete and asphalt surfaces
Ceramic[44]	100	Ceramic tiles
SDNET2018[45]	1,411	Building facades, bridges, sidewalks
Rissbilder[42]	2,736	Building surfaces (walls, bridges)
Crack500[21]	3,126	Pavement
GAPS384[46]	383	Pavement and concrete surfaces

In addition to CrackSeg9k, we also use two specific scene crack datasets: the close-range concrete crack dataset Ozgenel[47] and the low-altitude UAV-captured highway crack dataset Aerial Track[48], to further explore the generalization ability of CrackScopeNet. Among them, the image scene in the Ozgenel dataset is similar to the rock crack scene in CrackSeg9k, while the Aerial Track dataset includes post-earthquake highway crack images captured by UAVs, featuring predominantly small cracks amidst significant interference. As shown in Figure 3 (b) and (c), two randomly selected images from two datasets are displayed.

The Ozgenel dataset originally contains 458 high-definition images (4032x3024 pixels) with annotated cracks, collected from various buildings at Middle East Technical University. For our experiments, we crop these images into 448x448 pixel blocks, ensuring each block have at least 1% crack area. This process yields 2600 images, divided into 70% for training, 10% for validation, and 20% for testing.

The Aerial Track dataset comprises 4118 highway crack images (448x448 pixels) captured by UAVs after an earthquake. The dataset is divided into three parts: training, validation, and testing, with 2620, 598, and 900 images, respectively. We will transfer our models trained on CrackSeg9k to these two specific tasks.

3.2. Parameter Setting

3.2.1. Training and Fine-Tuning Configurations

Our experiments use the PaddleSeg[49] framework, performed on a desktop with an NVIDIA TITAN V GPU (12GB) and Ubuntu 20.04 with CUDA 10.1. To prevent overfitting, we adopt data augmentation methods such as random horizontal flipping, scaling (0.5 to 2), cropping, resizing, normalization, and random distortion to vary brightness, contrast, and saturation with a 50% probability.

For the CrackSeg9k dataset, specific training parameters are shown in Table 3. All training is conducted from scratch, without pre-training on other datasets. To manage the limited GPU memory, models with high memory usage (e.g., UNet, PSPNet) have their batch size and initial learning rate halved, while keeping the number of epochs unchanged, ensuring similar training effects.

 doi:10.20944/preprints202407.0705.v1

Table 3. The parameter settings for training on CrackSeg9k dataset.

Item	Setting
Epoch	200
Batch Size	16
Optimizer	Adamw
Weight decay	0.01
Beta1	0.9
Beta2	0.999
Initial learning rate	0.005
Learning rate decay type	poly
GPU memory	12 GB
Image size	400x400

Beyond CrackSeg9k, we transfer the model to the Ozgenel[38] and Aerial Track[48] datasets. During fine-tuning, we reduce the learning rate to 0.0001, limit epochs to 20, and adjust the batch size to 8. Input images are cropped to 448x448 for Ozgenel and 512x512 for Aerial Track.

3.2.2. Inference Configurations

To simulate edge model deployment on UAV-based platforms and evaluate inference speed without GPU acceleration, we deploy our models on drones equipped with a Raspberry Pi. The Raspberry Pi 4B, featuring a Broadcom BCM2711 CPU (1.5GHz), offers a low-power, cost-effective, and highly customizable computing platform. Due to its limited resources, high-FLOPs models like UNet, SegNet, and PSPNet are not deployed on the Raspberry Pi. Specifically, we use the Navio2 flight control board for flight control and data processing, which integrates seamlessly with the Raspberry Pi and includes an inertial measurement unit (IMU), barometer, and GPS module, providing a comprehensive flight control solution. The choice of the Raspberry Pi as the computing device is driven by its efficiency and suitability in resource-constrained environments, making it an ideal platform to validate model inference latency on actual low-computation resource devices.

3.3. Evaluation Metrics

Based on previous studies[29,50], We use four metrics to comprehensively evaluate model performance: precision (Pr), recall (Re), F1 score ($F1$), and mean Intersection over Union ($mIoU$). These indicators are defined as follows.

$$Pr = \frac{TP}{TP + FP} \tag{6}$$

$$Re = \frac{TP}{TP + FN} \tag{7}$$

$$F1 = \frac{2 * Pr * Re}{Pr + Re} \tag{8}$$

$$mIoU = mean(\frac{TP}{TP + FN + FP}) \tag{9}$$

Here, true positive (TP) represents correctly classified crack pixels, false positive (FP) represents background pixels incorrectly classified as crack categories, and false negative (FN) represents cracks incorrectly identified as background.

Next, evaluate the computational cost and complexity of the model using floating point operations ($FLOPs$) and parameters ($Params$). In addition, we use the average inference latency of a single image deployed on Navio2-based drones to evaluate the inference speed of lightweight models. A lightweight model suitable for drone platforms requires low parameter count, low FLOPS, and low inference latency.

 doi:10.20944/preprints202407.0705.v1

4. Experiment

In this section, we first conduct a comprehensive quantitative comparison between CrackScopeNet and the most advanced segmentation models in various metrics, visualize the results, and comprehensively analyze the detection performance. Subsequently, we explore the transfer learning capability of our model on crack datasets specific to other scenarios. Finally, we perform ablation studies to meticulously examine the significance and impact of each component within CrackScopeNet.

4.1. Comparative Experiments

The primary objective is to achieve an exceptional balance between the accuracy of crack region extraction and inference speed. Thus, we compare CrackScopeNet with three types of models: classical general semantic segmentation models, advanced lightweight semantic segmentation models, and the latest models designed explicitly for crack segmentation, totaling 13 models. Specifically, U-Net[51], PSPNet[37], SegNet[52], DeeplabV3+[23], SegFormer[19], and SegNext[24] are selected as six classical high-accuracy segmentation models. BiSeNet[28], BiSeNetV2[53], STDC[54], TopFormer[55], and SeaFormer[56] are chosen for their advantage in inference speed as lightweight semantic segmentation models. Notably, SegFormer, TopFormer, and SeaFormer are Transformer-based methods that have demonstrated outstanding performance on large datasets such as Cityscapes[57]. Additionally, we compare two specialized crack segmentation models, U2Crack[50] and HrSegNet[29], which have been optimized for the crack detection scenario based on general semantic segmentation models.

It is important to note that to ensure the models could be easily converted to ONNX format and deployed on edge devices with limited computational resources and memory, we select lightweight backbones: MobileNetV2[58] and ResNet-18[59] for the DeepLabV3+ and BiSeNet models, respectively. For SegFormer and SegNext, we choose the lightweight versions SegFormer-B0[19] and SegNext_MSCAN_Tiny[24], which are suited for real-time semantic segmentation as proposed by the authors. For TopFormer and SeaFormer, we discover during training that the tiny versions are difficult to converge, so we only utilize their base versions.

Quantitative Results. Table 4 presents the performance of each baseline network and the proposed CrackScopeNet on the CrackSeg9k dataset, with the best values highlighted in bold. Analyzing the accuracy of different types of segmentation networks in the table reveals that larger models generally achieve higher mIoU scores than lightweight models. Specifically, compared to classical high-accuracy models, the proposed CrackScopeNet achieves the best performance in terms of mIoU, Recall, and F1 scores. Although our model's precision is 1.26% lower than U-Net, U-Net has a poor recall performance (-2.24%), and our model's parameters and FLOPs are reduced by 12 and 48 times, respectively.

 doi:10.20944/preprints202407.0705.v1

Table 4. Performance of different methods and our method on the CrackSeg9k dataset.

Model	mIoU	Pr(%)	Re(%)	F1(%)	Params	FLOPs
Classical						
U-Net[51]	81.36	**90.60**	87.00	88.76	13.40M	75.87G
PSPNet[37]	81.69	89.19	88.72	88.95	21.06M	54.20G
SegNet[52]	80.50	89.71	86.57	88.11	29.61M	103.91G
DeepLabV3+[23]	80.96	88.555	88.29	88.42	2.76M	2.64G
SegFormer[19]	81.63	89.815	88.05	88.92	3.72M	4.13G
SegNext[24]	81.55	89.28	88.44	88.86	4.23M	3.72G
Lightweight						
BiSeNet[28]	81.01	89.74	87.26	88.48	12.93M	34.57G
BiSeNetV2[53]	80.66	89.36	87.11	88.22	2.33M	4.93G
STDC[54]	80.84	88.92	87.76	88.34	8.28M	5.22G
STDC2[54]	80.94	89.54	87.33	88.42	12.32M	7.26G
TopFormer[55]	80.96	89.28	87.60	88.43	5.06M	1.00G
SeaFormer[56]	79.13	87.29	87.19	87.20	4.01M	**0.64G**
Specific						
U2Crack[50]	81.45	90.125	87.52	88.80	1.19M	31.21G
HrSegNetB48[29]	81.07	90.39	86.78	88.55	5.43M	5.59G
HrSegNetB64[29]	81.28	90.44	87.03	88.70	9.65M	9.91G
CrackScopeNet	**82.15**	89.34	**89.24**	**89.29**	**1.047M**	**1.58G**

In terms of network lightweightness, the CrackScopeNet proposed in this paper achieves the best accuracy-lightweight balance on the CrackSeg9k dataset, as more intuitively illustrated in Figure1. Our model achieves the highest mIoU with only 1.047M parameters and 1.58G FLOPs, making it incredibly lightweight. CrackScopeNet's FLOPs are slightly higher than those of TopFormer and SeaFormer but lower than all other small models. Notably, due to the small size of crack dataset, the learning capability of lightweight segmentation networks is evidently limited, as mainstream lightweight segmentation models do not consider the unique characteristics of cracks, resulting in poor performance. The CrackScopeNet architecture, while maintaining superior segmentation performance, successfully achieves the design goal of a lightweight network structure, making it easily deployable on resource-constrained edge devices.

Moreover, compared to the state-of-the-art crack image segmentation algorithms, the proposed method achieves a mIoU score of 82.15% with only 1.047M parameters and 1.58G FLOPs, surpassing the highest-accuracy versions of U2Crack and HrSegNet models. Notably, the HrSegNet model employ an Online Hard Example Mining (OHEM) technique during training to improve accuracy. In contrast, we only use a cross-entropy loss function for model parameter updates without deliberately employing training tricks to enhance performance, showcasing the significant benefits of considering crack morphology in our model design.

Qualitative Results. Figures 4, 5, and 6 display the qualitative results of all models. CrackScopeNet achieves superior visual performance compared to other models. From the first, second, and third rows of Figure 4, it can be observed that CrackScopeNet and the more significant parameter segmentation algorithms achieve satisfactory results for high-resolution images with apparent crack features. In the fourth row, where the original image contains asphalt with color and texture similar to cracks, CrackScopeNet and SegFormer successfully overcome such background noise interference. This is attributed to their long-range contextual dependencies, effectively capturing relationships between cracks. From the fifth row, CrackScopeNet exhibits robust performance even under uneven illumination conditions. This can be attributed to the design of CrackScopeNet, which considers both local and global features of cracks, effectively suppressing other noises.

 doi:10.20944/preprints202407.0705.v1

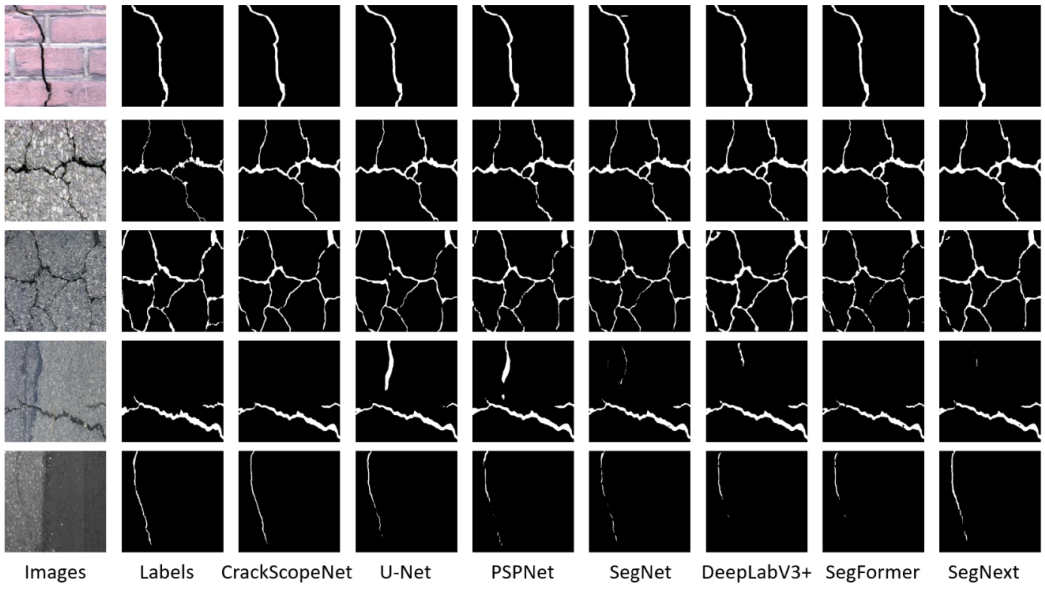

Figure 4. Visualization of the segmentation results of the classical segmentation models and our model on the CrackSeg9k test set.

Figure 5 clearly shows that lightweight networks struggle to eliminate background noise interference and produce fragmented segmentation results for fine cracks. This outcome is due to the limited parameters learned by lightweight models. Finally, Figure 6 presents the visualization results of the most advanced crack segmentation models. U2Crack[50], based on the ViT[17] architecture, achieves a broader receptive field, somewhat alleviating background noise but at the cost of significant computational overhead. HrSegNet[29] maintains a high-resolution branch to capture rich, detailed features. As seen in the last two columns of Figure 6, with increased channels in the HrSegNet network, more detailed information is extracted, but this also leads to misclassifying background information as cracks, explaining why HrSegNet's precision score is high while the recall score is low. In summary, CrackScopeNet outperforms other segmentation models with lower parameters and FLOPs by demonstrating excellent crack detection performance under various noise conditions.

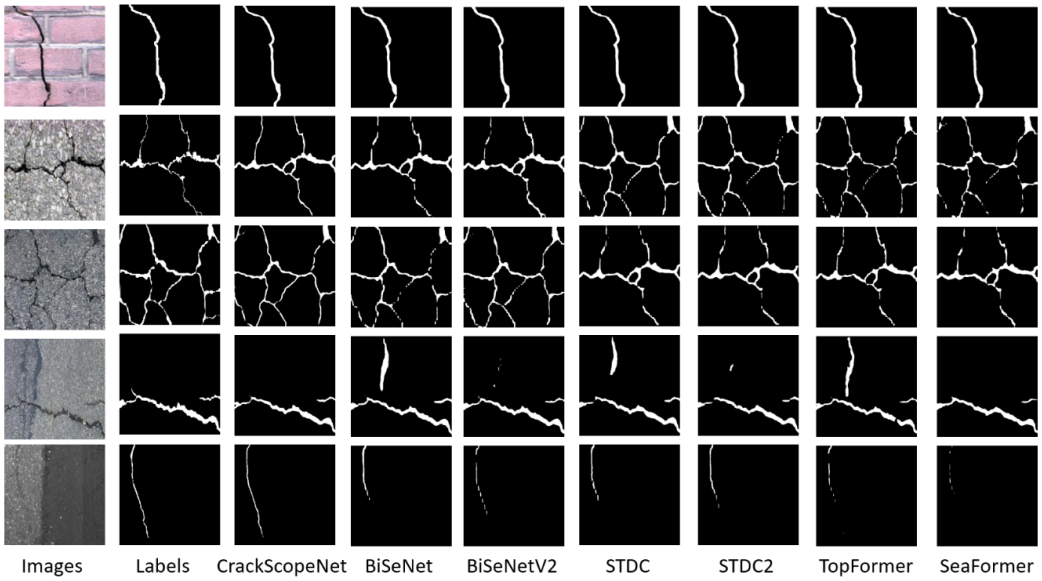

Figure 5. Visual segmentation results of the lightweight segmentation model and our model on the CrackSeg9k test set.

 doi:10.20944/preprints202407.0705.v1

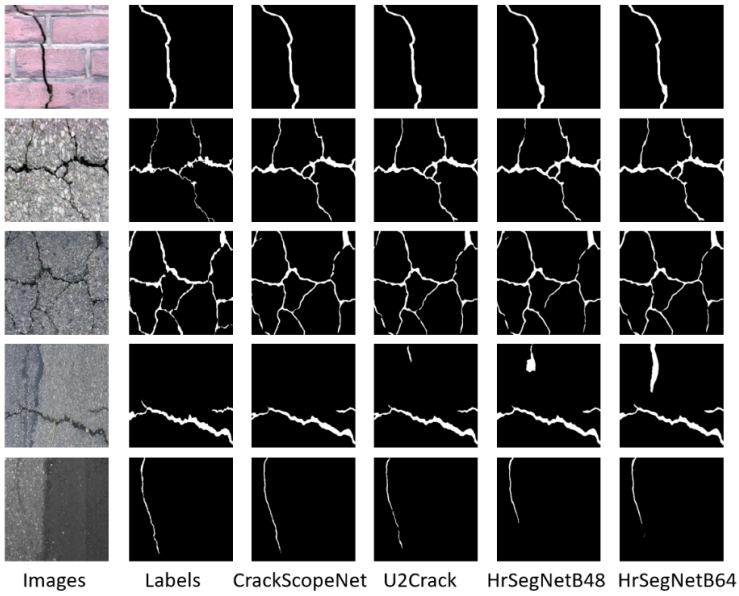

Figure 6. Visual segmentation results of the crack-specific segmentation model on the CrackSeg9k test set.

Inference on Navio2-based Drones.

In practical applications, there remains a substantial gap for real-time semantic segmentation algorithms designed and validated for mobile and edge devices, which face challenges such as limited memory resources and low computational efficiency. To better simulate edge devices used for outdoor structural health monitoring, we explore the inference speed of models without GPU acceleration. We convert the models to ONNX format and test the inference speed on Navio2-based drones equipped with a representative Raspberry Pi 4B, focusing on models with tiny FLOPs and parameter counts: BiSeNetV2, DeepLabV3+, STDC, HrSegNetB48, SegFormer, TopFormer, SeaFormer, and our proposed model. The test settings are input image size of 3×400×400, batch size of 1, and 2000 testing epochs. To ensure fair comparisons, we do not optimize or prune any models during deployment, meaning the actual inference delay in practical applications could be further reduced based on these test results.

As shown in Figure 7, the test results indicate that when running on highly resource-constrained drone platform, the proposed CrackScopeNet architecture achieves faster inference speed compared to other real-time or lightweight semantic segmentation networks based on convolutional neural networks, such as BiSeNet, BiSeNetV2, and STDC. Additionally, TopFormer and SeaFormer are designed with deployment on resource-limited edge devices in mind, resulting in extremely low inference latency. However, these two models perform poorly on the crack datasets due to inadequate data volume. Our proposed CrackScopeNet model, while maintaining rapid inference speed, achieves remarkable crack segmentation accuracy, establishing its advantage over competing models.

 doi:10.20944/preprints202407.0705.v1

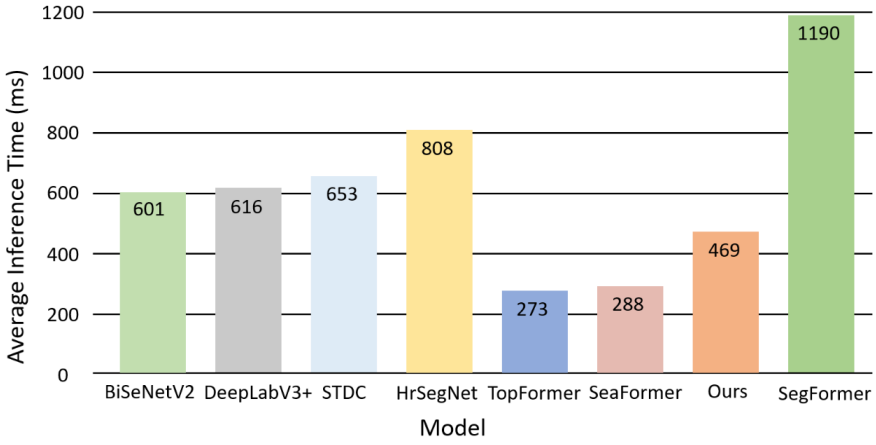

Figure 7. Results of inference speed test on Navio2-based Drones.

These results confirm the efficacy of deploying the CrackScopeNet model on outdoor mobile devices, where high-speed inference and lightweight architecture are crucial for the real-time processing and analysis of infrastructure surface cracks. By outperforming other state-of-the-art models, CrackScopeNet proves to be a suitable solution for addressing challenges associated with outdoor edge computing.

4.2. Scaling Study

To explore the adaptability of our model, we will adjust the number of channels and stack different numbers of Crack Scope Modules to cater to a broader range of application scenarios. Since CrackSeg9k is composed of multiple crack datasets, we will also investigate the model's transferability to specific application scenarios.

We adjust the base number of channels after the stem from 32 to 64. Correspondingly, the number of channels in the remaining three feature extraction stages increase from (32, 64, 128) to (64, 128, 160) to capture more features. Meanwhile, the number of Crack Scope Modules stacked in each stage is adjusted from (3, 3, 4) to (3, 3, 3). We refer to the adjusted model as CrackScopeNet_Large. First, we train CrackScopeNet_Large on CrackSeg9k in the same parameter settings as the base version and evaluate the model on the test set. Furthermore, we use the training parameters and weights obtained from CrackSeg9k for these two models as the basis for transferring the models to downstream tasks in two specific scenarios. Images in the Ozgenel dataset are cropped to 448x448 and are high-resolution concrete crack images, similar to some scenarios in CrackSeg9k. The Aerial Track Dataset consists of low-altitude drone-captured images of post-earthquake highway cracks, cropped to 512x512, a type of scene not present in CrackSeg9k.

Table 5 presents the mIoU scores, parameter counts, and FLOPs of the base model CrackScopeNet and the high-accuracy version CrackScopeNet_Large on the CrackSeg9k dataset and two specific scenario datasets. In this table, mIoU(F) represents the mIoU score obtained after pre-training the model on CrackSeg9k and fine-tuning it on the respective dataset. It is evident that the large version of the model achieves higher segmentation accuracy across all datasets, but with approximately double the parameters and three times the FLOPs. Therefore, if computational resources and memory are sufficient, and higher accuracy in crack segmentation is required, the large version or further stacking of Crack Scope Modules can be employed.

 doi:10.20944/preprints202407.0705.v1

Table 5. Evaluation results of the two versions of our model on three different datasets. CSNet and CSNet_L represent terms of CrackScopeNet and CrackScopeNet_Large. mIoU(F) represents the model mIoU score pre-trained on CrackSeg9k dataset.

Model	CrackSeg9k			Ozgenel			Aerial Track Dataset		
	mIoU	Param	FLOPs	mIoU	mIoU(F)	FLOPs	mIoU	mIoU(F)	FLOPs
CSNet	82.15%	1.05M	1.58G	90.05%	92.11%	1.98G	79.12%	82.63%	2.59G
CSNet_L	82.48%	2.20M	5.09G	90.71%	92.36%	6.38G	81.04%	83.43%	8.33G

For specific scenario training, whether from scratch or fine-tuning, our models are trained for only 20 epochs. It can be seen that even when training from scratch, our models converge quickly. We attribute this phenomenon to the initial design of CrackScopeNet, which consider the morphology of cracks and could successfully capture the necessary contextual information. For training using transfer learning, both versions of the model achieve remarkable mIoU scores on the Ozgenel dataset, with 90.1% and 92.31%, respectively. Even for the Aerial Track dataset, which includes low-altitude remote sensing images of highway cracks not seen in CrackSeg9k, our models still perform exceptionally well, achieving mIoU scores of 83.26% and 84.11%. These results demonstrate the proposed model's rapid adaptability to small datasets, aligning well with real-world tasks.

4.3. Diagnostic Experiments

To gain more insights into CrackScopeNet, a set of ablative studies on CrackSeg9k are conducted. All the methods mentioned in this section are trained with the same parameters for efficiency in 200 epochs.

Stripe-wise Context Attention. First, we examine the role of the critical SWA module in CrackScopeNet by replacing it with two advanced attention mechanisms, CBAM[36] and CA[35]. The results are shown in Table 6. It demonstrates that without any attention mechanism, merely stacking convolutional neural networks for feature extraction yields poor performance due to the limited receptive field. Then, the SWA attention mechanism, based on stripe pooling and one-dimensional convolution, is adopted, allowing the network structure to capture long-range contextual information. Under this configuration, the model exhibit the best performance. Figure 8 shows the class activation maps (CAM)[60] before the segmentation head of CrackScopeNet. It can be observed that without SWA, the model is easily disturbed by shadows, whereas with the SWA module, the model can focus on the global crack areas. Next, we sequentially replace the SWA module with the channel-spatial feature-based CBAM attention mechanism and the coordinate attention (CA) mechanism, which also uses stripe pooling. The model parameters do not change significantly, but the performance decline by 0.2% and 0.17%, respectively.

Table 6. Ablation study on effectiveness of each component in CrackScopeNet.

Attention			Mutil-branch	Decoder		mIoU(%)	FLOPs(G)
SWA	CA	CBAM		ours	ASPP		
				✓		81.34	1.57
	✓		✓	✓		81.98	1.58
		✓	✓	✓		81.95	1.58
✓				✓		81.91	1.61
✓			✓		✓	82.14	2.89
✓			✓	✓		82.15	1.58

 doi:10.20944/preprints202407.0705.v1

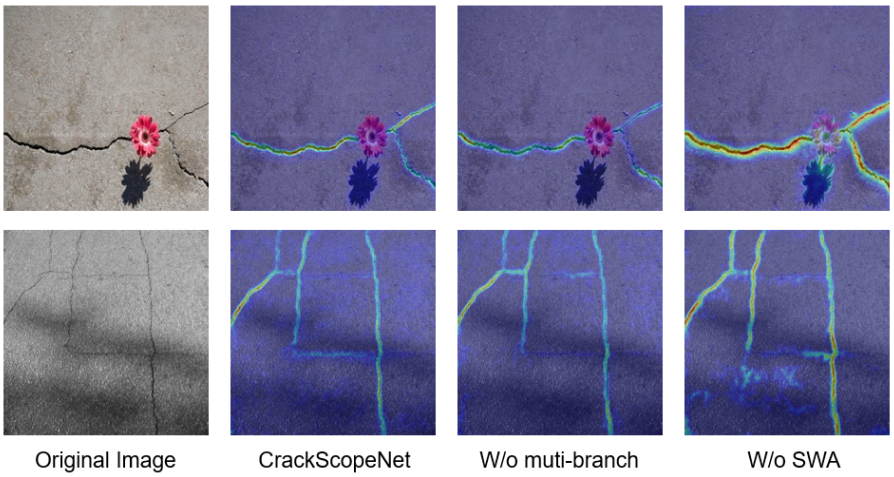

Figure 8. Showing visual explanations for different component of CrackScopeNet.

Furthermore, we explore the benefits of different attention mechanisms for other models by optimizing the advanced lightweight crack segmentation network HrSegNetB48[29]. HrSegNetB48 consists of high-resolution and auxiliary branches, merging shallow detail information with deep semantic information at each stage. Therefore, we add SWA, CBAM, and CA attention mechanisms after feature fusion to capture richer features. Table 6 shows the performance of HrSegNetB48 with different attention mechanisms, clearly indicating that introducing the SWA attention mechanism to capture long-range contextual information provides the most significant benefit.

Table 7. Adding different attention mechanisms to HrSegNet.

model	mIoU(%)	Pr(%)	Re(%)	F1(%)	Params	FLOPs
HrSegNetB48	81.07	90.39	86.78	88.55	5.43M	5.59G
HrSegNetB48+CBAM	81.16	90.40	86.90	8.61	5.44M	5.60G
HrSegNetB48+CA	81.20	90.24	87.08	88.63	5.44M	5.60G
HrSegNetB48+SWA	81.72	89.65	88.33	88.98	5.48M	5.60G

Multi-scale Branch. Then, we examine the effect of the Multi-scale Branch in our Crack Scope Module. To ensure fairness, we replace the multi-scale branch with a convolution of larger kernel size, 5x5 instead of 3x3. The results with or without the multi-scale branch are shown in Table 6. It is evident that using a 5x5 kernel size convolution instead of the multi-scale branch, even with more floating-point computations, decreases the mIoU score (-0.16%). This demonstrates that blindly adopting large kernel convolutions increases computational overhead without significant performance improvement. The benefits brought by multi-scale branch are further analyzed through the CAM. As shown in the third column of Figure 8, when multi-scale branch is not used, it is obvious that the network misses the feature information of small cracks, while our model can perfectly capture the features of cracks of various shapes and sizes.

Decoder. CrackScopeNet uses a simple decoder to fuse feature information of different scales, complete the compression of channel features and the fusion of features at different stages. At present, the most popular decoders combine Atrous Spatial Pyramid Pooling (ASPP)[23] module to introduce multi-scale information. In order to explore whether the introduction of ASPP module can bring benefits to our model and whether our proposed lightweight decoder is effective, we replace decoder with the ASPP method adopted by DeepLabV3+[23], and the results are shown in the last two rows of Table 6. It can be seen that the computational overhead is large because of the need to perform parallel dilated convolution operations on deep semantic information, but the performance of the model is not improved. We believe that this is because local feature information and long-distance context

 doi:10.20944/preprints202407.0705.v1

information have been taken into account when feature extraction is carried out in each stage, so it does not need to be complicated for the design of decoder.

4.4. Experiment Conclusion

Based on the comparative experiments on performance, parameter count, and FLOPs conducted in previous sections, CrackScopeNet demonstrates significant advantages over both classical and lightweight semantic segmentation models. On the composite CrackSeg9k dataset, CrackScopeNet achieves high segmentation accuracy and shows excellent transferability to specific scenarios. Notably, it maintains a low parameter count and minimal FLOPs, which translates to low-latency inference speeds on resource-constrained drone platforms without the need for GPU acceleration. This efficiency is achieved by considering crack morphology characteristics, allowing CrackScopeNet to remain lightweight and computationally efficient. This makes it particularly suitable for deployment on mobile devices in outdoor environments. In summary, CrackScopeNet achieves a better balance between segmentation accuracy and inference speed compared to other networks examined in this study, making it a promising solution for timely crack detection and analysis using drones in infrastructure surfaces.

5. Discussion

In this paper, we present CrackScopeNet, a lightweight infrastructure surface crack segmentation network specifically designed to address the challenges posed by varying crack sizes, irregular contours, and subtle differences between cracks and normal regions in real-world applications. The proposed CrackScopeNet network structure captures the local context information and long-distance dependencies of cracks through a lightweight multi-scale branch and SWA attention mechanism, respectively, and effectively extracts the low-level details and high-level semantic information required for accurate crack segmentation.

Experimental results demonstrate that CrackScopeNet delivers robust performance and high accuracy. It outperforms larger models like SegFormer in terms of efficiency, significantly reducing the number of parameters and computational cost. Furthermore, CrackScopeNet achieves faster inference speeds than other lightweight models such as BiSeNet and STDC, even without GPU acceleration. This makes it highly suitable for deployment on resource-constrained drone platforms, enabling efficient and low-latency crack detection in structural health monitoring. By making the model and code publicly available, we aim to advance the application of UAV remote sensing technology in infrastructure maintenance, providing an efficient and practical tool for the timely detection and analysis of cracks.

Unfortunately, in this era of large models, our model has only been trained and evaluated on datasets containing a few thousand images, while a large amount of data collection and huge manual labeling is the bottleneck. Recent advances in generative AI and self-supervised learning bypass the limitations imposed by data acquisition and manual annotation. Researchers use the inherent structure or attributes of existing data to generate richer "fake images" and "fake labels", and it will be a very interesting research avenue to apply them to crack detection.

Author Contributions: T.Z. designed the architecture and the comparative experiments, wrote the manuscript; L.Q. revised the manuscript and assisted T.Z. in conducting the experiments; Q.Z. and H.Z. made suggestion for the experiments and assisted in revising the manuscript; L.Z. and R.W. conducted investigation and code testing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by research on identification and variation analysis methods for rock fractures, development of a real-time monitoring model for falling rocks based on machine vision, research project on hazard warning algorithm and terminal equipment for rock collapse based on vibration data of Chongqing Institute of Geology and Mineral Resources, grant number F2023304, F2023045 and cstc2022jxjl00006;

Data Availability Statement: The code and data that support the findings of this study are available on GitHub at https://github.com/ttkingzz/CrackScopeNet.

Acknowledgments: The authors would like to thank the editors and the reviewers for their valuable suggestions.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Minh Dang, L.; Wang, H.; Li, Y.; Nguyen, L.Q.; Nguyen, T.N.; Song, H.K.; Moon, H. Deep learning-based masonry crack segmentation and real-life crack length measurement. *359*, 129438. doi:10.1016/j.conbuildmat.2022.129438.
2. Zheng, M.; Lei, Z.; Zhang, K. Intelligent detection of building cracks based on deep learning. *103*, 103987. doi:10.1016/j.imavis.2020.103987.
3. Ha, J.; Kim, D.; Kim, M. Assessing severity of road cracks using deep learning-based segmentation and detection. *78*, 17721–17735. Number: 16 Publisher: Springer, doi:10.1007/s11227-022-04560-x.
4. Zhang, J.; Qian, S.; Tan, C. Automated bridge surface crack detection and segmentation using computer vision-based deep learning model. *115*, 105225. doi:10.1016/j.engappai.2022.105225.
5. Deng, J.; Singh, A.; Zhou, Y.; Lu, Y.; Lee, V.C.S. Review on computer vision-based crack detection and quantification methodologies for civil structures. *356*, 129238. doi:10.1016/j.conbuildmat.2022.129238.
6. Gavilán, M.; Balcones, D.; Marcos, O.; Llorca, D.F.; Sotelo, M.A.; Parra, I.; Ocaña, M.; Aliseda, P.; Yarza, P.; Amírola, A. Adaptive Road Crack Detection System by Pavement Classification. *11*, 9628–9657. Number: 10 Publisher: Molecular Diversity Preservation International, doi:10.3390/s111009628.
7. Jahanshahi, M.R.; Jazizadeh, F.; Masri, S.F.; Becerik-Gerber, B. Unsupervised Approach for Autonomous Pavement-Defect Detection and Quantification Using an Inexpensive Depth Sensor. Publisher: American Society of Civil Engineers.
8. Zhang, D.; Zou, Q.; Lin, H.; Xu, X.; He, L.; Gui, R.; Li, Q. Automatic pavement defect detection using 3D laser profiling technology. *96*, 350–365. doi:10.1016/j.autcon.2018.09.019.
9. Iyer, S.; Sinha, S.K. Segmentation of Pipe Images for Crack Detection in Buried Sewers. *21*, 395–410. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8667.2006.00445.x, doi:10.1111/j.1467-8667.2006.00445.x.
10. Sun, B.C.; Qiu, Y.j. Automatic Identification of Pavement Cracks Using Mathematic Morphology | Proceedings | Vol , No.
11. Kamaliardakani, M.; Sun, L.; Ardakani, M.K. Sealed-Crack Detection Algorithm Using Heuristic Thresholding Approach. *30*, 04014110. Publisher: American Society of Civil Engineers, doi:10.1061/(ASCE)CP.1943-5487.0000447.
12. Mohan, A.; Poobal, S. Crack detection using image processing: A critical review and analysis. *57*, 787–798. doi:10.1016/j.aej.2017.01.020.
13. Qu, Z.; Lin, L.D.; Guo, Y.; Wang, N. An improved algorithm for image crack detection based on percolation model. *10*, 214–221. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/tee.22056, doi:10.1002/tee.22056.
14. Cha, Y.J.; Ali, R.; Lewis, J.; Buyukozturk, O. Deep learning-based structural health monitoring. *161*, 105328. doi:10.1016/j.autcon.2024.105328.
15. Liu, Z.; Cao, Y.; Wang, Y.; Wang, W. Computer vision-based concrete crack detection using U-net fully convolutional networks. *104*, 129–139. doi:10.1016/j.autcon.2019.04.005.
16. Yang, J.; Wang, W.; Lin, G.; Li, Q.; Sun, Y.; Sun, Y. Infrared Thermal Imaging-Based Crack Detection Using Deep Learning. *7*, 182060–182077. Conference Name: IEEE Access, doi:10.1109/ACCESS.2019.2958264.
17. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; Houlsby, N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR* **2021**.
18. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. Proceedings of the IEEE/CVF international conference on computer vision, pp. 10012–10022.
19. Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J.M.; Luo, P. SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. Advances in Neural Information Processing Systems. Curran Associates, Inc., Vol. 34, pp. 12077–12090.
20. Lin, Q.; Li, W.; Zheng, X.; Fan, H.; Li, Z. DeepCrackAT: An effective crack segmentation framework based on learning multi-scale crack features. *126*, 106876. doi:10.1016/j.engappai.2023.106876.

21. Yang, F.; Zhang, L.; Yu, S.; Prokhorov, D.; Mei, X.; Ling, H. Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection. *21*, 1525–1535. Conference Name: IEEE Transactions on Intelligent Transportation Systems, doi:10.1109/TITS.2019.2910595.
22. Chu, H.; Wang, W.; Deng, L. Tiny-Crack-Net: A multiscale feature fusion network with attention mechanisms for segmentation of tiny cracks. *37*, 1914–1931. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/mice.12881, doi:10.1111/mice.12881.
23. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. Proceedings of the European conference on computer vision (ECCV), pp. 801–818.
24. Guo, M.H.; Lu, C.Z.; Hou, Q.; Liu, Z.; Cheng, M.M.; Hu, S.m. SegNeXt: Rethinking Convolutional Attention Design for Semantic Segmentation. *35*, 1140–1156.
25. Duan, Z.; Liu, J.; Ling, X.; Zhang, J.; Liu, Z. ERNet: A Rapid Road Crack Detection Method Using Low-Altitude UAV Remote Sensing Images. *16*, 1741. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute, doi:10.3390/rs16101741.
26. Meng, S.; Gao, Z.; Zhou, Y.; He, B.; Djerrad, A. Real-time automatic crack detection method based on drone. *38*, 849–872. Publisher: John Wiley & Sons, Ltd, doi:10.1111/mice.12918.
27. Paszke, A.; Chaurasia, A.; Kim, S.; Culurciello, E. ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation, [1606.02147 [cs]].
28. Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; Sang, N. Bisenet: Bilateral segmentation network for real-time semantic segmentation. Proceedings of the European conference on computer vision (ECCV), pp. 325–341.
29. Li, Y.; Ma, R.; Liu, H.; Gaoli, C. Real-time high-resolution neural network with semantic guidance for crack segmentation. *156*, 105112. Publisher: Elsevier, doi:10.1016/j.autcon.2023.105112.
30. Liu, Z.; Mao, H.; Wu, C.Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A convnet for the 2020s. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 11976–11986.
31. Yu, W.; Luo, M.; Zhou, P.; Si, C.; Zhou, Y.; Wang, X.; Feng, J.; Yan, S. Metaformer is actually what you need for vision. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10819–10829.
32. Cai, X.; Lai, Q.; Wang, Y.; Wang, W.; Sun, Z.; Yao, Y. Poly kernel inception network for remote sensing detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 27706–27716.
33. Ding, X.; Zhang, X.; Han, J.; Ding, G. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 11963–11975.
34. Tang, Y.; Han, K.; Guo, J.; Xu, C.; Xu, C.; Wang, Y. GhostNetv2: Enhance cheap operation with long-range attention. *35*, 9969–9982.
35. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 13713–13722.
36. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. Proceedings of the European conference on computer vision (ECCV), pp. 3–19.
37. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. pp. 2881–2890.
38. Kulkarni, S.; Singh, S.; Balakrishnan, D.; Sharma, S.; Devunuri, S.; Korlapati, S.C.R. CrackSeg9k: A Collection and Benchmark for Crack Segmentation Datasets and Frameworks. Computer Vision – ECCV 2022 Workshops; Karlinsky, L.; Michaeli, T.; Nishino, K., Eds.; Springer Nature Switzerland: Cham, 2023; pp. 179–195.
39. Dais, D.; Bal, E.; Smyrou, E.; Sarhosis, V. Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning. *125*, 103606. doi:10.1016/j.autcon.2021.103606.
40. Shi, Y.; Cui, L.; Qi, Z.; Meng, F.; Chen, Z. Automatic Road Crack Detection Using Random Structured Forests. *17*, 3434–3445. Conference Name: IEEE Transactions on Intelligent Transportation Systems, doi:10.1109/TITS.2016.2552248.
41. Zou, Q.; Cao, Y.; Li, Q.; Mao, Q.; Wang, S. CrackTree: Automatic crack detection from pavement images. *33*, 227–238. doi:10.1016/j.patrec.2011.11.004.

42. Pak, M.; Kim, S. Crack Detection Using Fully Convolutional Network in Wall-Climbing Robot. Advances in Computer Science and Ubiquitous Computing; Park, J.J.; Fong, S.J.; Pan, Y.; Sung, Y., Eds.; Springer Singapore: Singapore, 2021; pp. 267–272.
43. Liu, Y.; Yao, J.; Lu, X.; Xie, R.; Li, L. DeepCrack: A deep hierarchical feature learning architecture for crack segmentation. *338*, 139–153. doi:10.1016/j.neucom.2019.01.036.
44. Junior, G.S.; Ferreira, J.; Millán-Arias, C.; Daniel, R.; Junior, A.C.; Fernandes, B.J.T. Ceramic Cracks Segmentation with Deep Learning. *11*, 6017. Number: 13 Publisher: Multidisciplinary Digital Publishing Institute, doi:10.3390/app11136017.
45. Dorafshan, S.; Thomas, R.J.; Maguire, M. SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. *21*, 1664–1668. doi:10.1016/j.dib.2018.11.015.
46. Eisenbach, M.; Stricker, R.; Seichter, D.; Amende, K.; Debes, K.; Sesselmann, M.; Ebersbach, D.; Stoeckert, U.; Gross, H.M. How to get pavement distress detection ready for deep learning? A systematic approach. 2017 International Joint Conference on Neural Networks (IJCNN), pp. 2039–2047. ISSN: 2161-4407, doi:10.1109/IJCNN.2017.7966101.
47. Özgenel, F. Concrete Crack Segmentation Dataset. doi:10.17632/jwsn7tfbrp.1.
48. Hong, Z.; Yang, F.; Pan, H.; Zhou, R.; Zhang, Y.; Han, Y.; Wang, J.; Yang, S.; Chen, P.; Tong, X.; Liu, J. Highway Crack Segmentation From Unmanned Aerial Vehicle Images Using Deep Learning. *19*, 1–5. Conference Name: IEEE Geoscience and Remote Sensing Letters, doi:10.1109/LGRS.2021.3129607.
49. Liu, Y.; Chu, L.; Chen, G.; Wu, Z.; Chen, Z.; Lai, B.; Hao, Y. PaddleSeg: A High-Efficient Development Toolkit for Image Segmentation, 2021, [arXiv:cs.CV/2101.06175].
50. Shi, P.; Zhu, F.; Xin, Y.; Shao, S. U2CrackNet: a deeper architecture with two-level nested U-structure for pavement crack detection. *22*, 2910–2921. Publisher: SAGE Publications, doi:10.1177/14759217221140976.
51. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015; Navab, N.; Hornegger, J.; Wells, W.M.; Frangi, A.F., Eds. Springer International Publishing, pp. 234–241. doi:10.1007/978-3-319-24574-4_28.
52. Badrinarayanan, V.; Kendall, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *39*, 2481–2495. Publisher: IEEE.
53. Yu, C.; Gao, C.; Wang, J.; Yu, G.; Shen, C.; Sang, N. BiSeNet V2: Bilateral Network with Guided Aggregation for Real-Time Semantic Segmentation. *129*, 3051–3068. doi:10.1007/s11263-021-01515-2.
54. Fan, M.; Lai, S.; Huang, J.; Wei, X.; Chai, Z.; Luo, J.; Wei, X. Rethinking BiSeNet for Real-Time Semantic Segmentation. pp. 9716–9725.
55. Zhang, W.; Huang, Z.; Luo, G.; Chen, T.; Wang, X.; Liu, W.; Yu, G.; Shen, C. TopFormer: Token Pyramid Transformer for Mobile Semantic Segmentation. pp. 12083–12093.
56. Wan, Q.; Huang, Z.; Lu, J.; Yu, G.; Zhang, L. SeaFormer: Squeeze-enhanced Axial Transformer for Mobile Semantic Segmentation. International Conference on Learning Representations (ICLR), 2023.
57. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The cityscapes dataset for semantic urban scene understanding. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3213–3223.
58. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520.
59. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.
60. Zhou, B.; Khosla, A.; A., L.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. *CVPR* **2016**.