DOI: 10.1111/mice.12918

RESEARCH ARTICLE

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING WILEY

Real-time automatic crack detection method based on drone

Shiqiao Meng[1] | **Zhiyuan Gao**[2] | **Ying Zhou**[1] | **Bin He**[2] | **Abderrahim Djerrad**[1]

[1]State Key Laboratory of Disaster Reduction in Civil Engineering, Tongii University, Shanghai, China

[2]Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, China

Correspondence
Ying Zhou, State Key Laboratory of Disaster Reduction in Civil Engineering, Tongii University, Shanghai 200092, China.
Email: yingzhou@tongji.edu.cn

Present address
Ying Zhou, No. 1239, Siping Road, Yangpu District, Shanghai, China.

Funding information
National Natural Science Foundation of China, Grant/Award Number: 52025083

Abstract

Real-time automated drone-based crack detection can be used for efficient building damage assessment. This paper proposes an automated real-time crack detection method based on a drone. Using a lightweight classification algorithm, a lightweight segmentation algorithm, a high-precision segmentation algorithm, and a crack width measurement algorithm, the cracks are classified, roughly segmented, finely segmented, and the maximum width is extracted. A crack information-assisted drone flight automatic control algorithm for automatic crack detection guides the drone toward the crack position. The effectiveness of the crack detection algorithm and the crack information-assisted drone flight automatic control algorithm was tested on two different datasets, a two-story building, and a 16-m-high shaking table test building. The results show that crack detection can be finished in real-time during the flight. Using the proposed method can significantly improve the MIoU of crack edge detection and the accuracy of maximum crack width measurement under the non-ideal shooting conditions of the actual inspection situation by automatically approaching the vicinity of the crack.

1 | INTRODUCTION

Detecting cracks is a vital part of structural health monitoring (Sirca & Adeli, 2018). When a building, such as a reinforced concrete structure, develops too many cracks, the occurrence of cracks reduces the integrity and rigidity of the structure, thereby affecting the safety of the entire structure, and may cause a huge loss of life and property. In addition, when an earthquake disaster occurs, crack detection on large-scale damaged buildings can provide critical decision-making information for damage assessment and maintenance. Therefore, it is necessary to examine the morphological change and development of the crack by measuring the crack's state and evaluating the degree of the influence of the crack on the structure. Crack detection is of great significance to the daily safety maintenance of buildings, the rapid assessment of post-disaster damage to buildings, and the prevention of loss of life and property.

Due to the strong visual characteristics of crack diseases, many researchers apply image detection technology to crack detection, including the morphological image processing method (Iyer & Sinha, 2006; Sun & Qiu, 2007), the filtering method (Alaknanda et al., 2009; Kamaliardakani et al., 2016; Mohan & Poobal, 2018), the percolation model-based method (Qu et al., 2015), and the foreground–background separation technique for crack detection (Nayyeri et al., 2019). However, the generalization performance of such algorithms is limited. In the actual engineering environment, there is much debris around the cracks, so the detection results are seriously affected by the noise in the surrounding environment.

The convolutional neural network (CNN) in deep learning technology has brought breakthrough progress that can significantly improve the computer's ability to classify images while reducing the impact of noise in the environment on prediction results. Based on a CNN combined

with the naive Bayes data fusion scheme, a method of using CNN to analyze a single video frame for crack detection was proposed by F.-C. Chen and Jahanshahi (2018). Cha et al. (2018) used region-based CNN (R-CNN) to detect various concrete diseases, such as cracks and rust, and built a framework model for obvious diseases. Deng et al. (2020) realized the accurate object detection of concrete cracks with crack-like handwriting scripts in the environment through the faster R-CNN algorithm. Y. Zhang and Yuen (2021) used a fusion features-based broad learning system to achieve efficient training and prediction of the crack detection model. Zou et al. (2022) realized the object detection of crack and safety evaluation method of the reinforced concrete structure after earthquake disaster through the v4 (YOLO v4) algorithm. However, the object detection algorithms are unable to obtain the geometric edge of the crack, implying that they are unable to quantify the geometric size of the crack.

In terms of edge detection of cracks, the development of semantic segmentation research (L.-C. Chen et al., 2018; Long et al., 2015; Ronneberger et al., 2015) makes it possible to use computer vision techniques to accurately perform crack edge detection. The current research on crack edge detection based on deep learning is mainly to improve the methods based on a fully convolutional network (FCN; Dung & Anh, 2019; Li et al., 2019; Ye et al., 2019), a self-attention mechanism (Y. Pan et al., 2020; Song et al., 2019), CNN (Fei et al., 2019; S.-Y. Kong et al., 2021; Ni et al., 2019; A. Zhang et al., 2017), UNet (J. Chen & He, 2022), recurrent neural network (RNN; A. Zhang et al., 2018), and DeepLab (Meng et al., 2020). Besides, some researchers utilize multi-stage crack detection and achieve high-precision crack edge detection by combining the classification algorithm with the segmentation algorithm (J. Liu et al., 2020; Meng et al., 2022). C. Liu and Xu (2022) used an image feature point matching technology and image-to-image conversion technology to standardize the pavement crack image at night. The detection of pavement crack images at night can be realized without re-labeling the dataset. Çelik and König (2022) proposed a sigmoid-optimized encoder–decoder network for crack segmentation with copyedit-paste transfer learning to improve detection accuracy. Compared with the traditional edge detection algorithms, such as the Canny algorithm (Zhao et al., 2010), these methods are more robust in crack edge detection and significantly improve the continuity of segmentation results (J. Liu et al., 2020). However, although the above-mentioned methods can realize high-precision crack detection under various conditions, due to the complexity of these models and the amount of calculation, most of the inference speeds cannot achieve dozens of predictions per second.

In terms of the crack width measurement, Y. Zhou and Liu (2019) achieved high-precision automatic detection of crack width through morphological processing, connected component labeling algorithm and skeleton extraction algorithm. Under the ideal shooting conditions in the laboratory, the absolute error of crack width measurement by this method is 0.1 mm. Yang et al. (2018) obtained the edge of the crack through FCN and then obtained the length of the crack through the skeleton line extraction algorithm. They obtained this by integrating the geometric correction index of the detected area and dividing it by the length. Jin et al. (2020) used the medial axis transform (MAT) algorithm and the pruning algorithm to extract the crack skeleton line, and then used the algorithm based on the flexible kernel to calculate the width of the crack. Wang et al. (2018) introduced the Laplace equation and used a crack blob extraction algorithm and a crack boundary extraction algorithm to measure the millimeter-level crack width of the crack image under ideal conditions. However, since most research use datasets under ideal shooting conditions in experiments, it is difficult to achieve very high accuracy in practical situations where the shooting distance and shooting conditions are uncertain.

In order to automate crack detection, it is necessary to implement crack detection algorithms on multiple hardware devices. Furthermore, since it is difficult to manually collect image information for infrastructures such as high-rise structures and bridges, the use of large inspection equipment for image acquisition is more advantageous in terms of efficiency and safety. Due to their good maneuverability and wide detection range, drones have become good hardware support for crack detection problems (Yoon et al., 2018). In addition, for large-scale detection range, such as rapid crack detection for a large number of buildings after earthquake disasters, using a drone as hardware can give full play to its advantages and achieve more efficient detection. Choi and Kim (2015) used a drone to capture images and utilize the edge detection algorithm to detect cracks on building surfaces in a simple environment. Kang and Cha (2018) proposed an autonomous drone method using ultrasonic beacons to replace the role of global positioning system (GPS), a deep CNN for damage detection, and a geo-tagging method for the localization of damage. Jiang and Zhang (2020) used a wall-climbing unmanned aerial system to detect cracks on the building surface and developed an Android application for fast crack data acquisition.

Researchers have also conducted some research on damage detection of post-earthquake buildings. X. Kong and Li (2018) used a feature tracking technology to extract features from video streams for fatigue cracks on steel bridges and proposed a crack detection and localization algorithm

based on feature points. Liang (2019) proposed a three-level image-based approach for post-disaster inspection of the reinforced concrete bridge using deep learning with novel training strategies. X. Pan and Yang (2020) used the YOLOv2 model to classify the damage degree of reinforced concrete buildings and performed repair cost evaluation on the buildings based on the classification results.

To summarize, several researchers are currently researching crack detection algorithms and have proposed numerous effective crack detection algorithms for different situations. However, despite their high accuracy, most of the algorithms are not suitable for deployment on drones for real-time crack detection due to the limitation of model parameters and the huge computational cost. Especially in the case of large-scale rapid crack detection, such as after an earthquake disaster, there is an urgent need for a real-time and relatively high-precision crack detection method. The crack width measurement needs to reach the millimeter level in the structural health monitoring of concrete structures and rapid post-disaster damage assessment. However, it is difficult to achieve millimeter accuracy for the crack width obtained by shooting with the camera of a drone far away from the building. In addition, most studies on crack width measurement only use pictures under ideal shooting conditions. It is difficult to calibrate the shooting distance for crack detection based on drones in practical situations. Besides, the drone has turbulence in flight, and there are many environmental interference factors in the captured images. Therefore, it is necessary to realize accurate crack width measurement based on drones under complex practical conditions. Furthermore, to ensure drone flight safety, it is difficult to control the drone to maintain a short distance from the building for a long time through fixed-point navigation or manual operation of the drone, especially in large-scale crack detection. Therefore, most methods need to keep the drone at a certain distance from the building, which will lead to a decrease in detection accuracy.

To address these problems, this paper proposes a crack detection process that can be used in drones. Through the crack detection method proposed in this paper, the drone can automatically identify cracks and automatically approach the vicinity of cracks only when needed to achieve more accurate and efficient crack detection. The method includes a lightweight crack classification algorithm, a lightweight crack segmentation algorithm, and a high-precision crack segmentation algorithm in the crack edge detection part. It can be combined with the depth map calculated by the binocular camera to provide a real-time decision-making basis for drone flight control while significantly improving the accuracy and efficiency of crack geometry edge extraction. In addition, by combining the edge information of the crack and the depth map calculated by the binocular camera, this paper proposes a maximum crack width measurement algorithm to achieve the maximum crack width extraction based on the drone under non-ideal shooting conditions. Furthermore, this paper proposes a crack information-assisted drone flight automatic control algorithm, which uses the crack information obtained from the crack detection algorithm to assist the drone in automated path planning. It can ensure that the drone automatically approaches the crack area while ensuring stable flight of the drone. Thus, the millimeter-level cracks width measurement can be realized by combining the crack information-assisted drone flight automatic control algorithm and the proposed crack detection algorithms.

To verify the effectiveness of the crack detection method, the lightweight classification algorithm, the lightweight segmentation algorithm, and the high-precision segmentation algorithm have been tested on two datasets, SBGCrack and METU-Crack, respectively. Furthermore, to verify the effectiveness of the crack information-assisted drone flight automatic control algorithm and the whole maximum crack width measurement process in the actual situation, the flight trajectory of the drone and the accuracy of crack width measurement were tested in a building of Tongji University. Finally, to verify the effectiveness of the whole maximum crack width measurement process, a DJI Matrix300 drone equipped with a binocular camera and an Nvidia Jetson AGX Xavier was used to conduct experiments for a 16-m height shaking table test building. The remainder of this paper consists of Section 2—methodology, Section 3—experiments, and Section 4—conclusion.

2 | METHODOLOGY

The method proposed in this paper is mainly applied in drone-based automatic real-time crack detection. The information from crack detection can be used to assist drones in path planning to achieve higher-precision crack geometry edge extraction, more accurate maximum crack width measurement, and obtain the spatial position of the crack.

2.1 | The overall process of crack detection

In the crack detection process, when the drone has been kept close to the building for a long time, the probability of a collision is extremely high. Therefore, the drone needs to maintain a relatively long distance from the building during normal navigation, which makes high-precision crack

detection difficult to achieve. Using the crack detection method in this paper, when the drone finds a crack, it can approach a position extremely close to the crack smoothly, thereby achieving higher precision and higher efficiency crack detection. Since all operations are carried out on the drone, the detection of cracks can be completed during the drone's navigation without waiting for the end of image acquisition. Therefore, the method in this paper can realize real-time crack detection since the crack detection results can be quickly obtained during the drone's flight. In addition, the definition of automated crack detection in this paper is that the drone can automatically and smoothly navigate to the vicinity of the crack through the real-time rough detection result, to achieve higher-precision crack edge detection and maximum crack width measurement. The crack detection method proposed in this paper mainly aims at cracks with a width of centimeters and millimeters since such cracks strongly affect the safety of buildings after an earthquake. Sub-millimeter cracks, which have less impact on structures and are difficult to detect, are not the focus of crack detection in this paper. The crack detection process used in this paper consists of three parts.

In the first part of the crack detection process, when the drone detects cracks, the image of the binocular camera is obtained and downsampled. Then a lightweight crack classification model image is used to classify the image to determine whether there is a crack in the image. If the classification result is that there are cracks in the image, use the lightweight crack classification algorithm to process the resized image to obtain a rough crack area. By combining the crack position and the depth image generated by the binocular camera, the coordinates of the crack in the drone coordinate system can be calculated, thereby providing critical information for the crack information-assisted drone flight automatic control algorithm. Since only when the crack information-assisted drone flight automatic control algorithm obtains real-time crack position information can the drone be effectively controlled to move to the vicinity of the crack. The crack detection algorithm inevitably has an error rate. However, if the overlapping multi-frame images are repeatedly detected to make full use of the information in the captured images, the probability of missing cracks can be significantly reduced.

This paper defines real time as the image processing speed that can fully utilize the captured images and provide critical information for the drone path planning in time. Since the existing captured image information must be fully utilized, the processing speed of the lightweight crack classification algorithm and the lightweight crack segmentation algorithm must exceed the camera's frame rate, which can fully utilize the existing information. This means that these two algorithms need to process dozens of images in one second. Moreover, crack classification is simpler than crack segmentation, and the accuracy of crack classification algorithms is generally higher. Therefore, using a lightweight crack classification algorithm to preprocess images before crack segmentation can significantly reduce the probability of misjudging the existence of cracks. To ensure that this process can be performed in real time, the input images are downsampled to reduce the computational and memory requirements of the lightweight crack detection algorithm.

The second part of the crack detection process is achieved by the crack information-assisted drone flight automatic control algorithm proposed in this paper. By using the crack information-assisted drone flight automatic control algorithm, the drone will automatically approach the crack and hover smoothly in the vicinity of the crack. The algorithm proposed in this paper sets an expected drone position based on the actual location of the crack and makes the drone automatically move in that direction. The automatic control algorithm is successfully completed once the drone flies to the target position and hovers smoothly. When the control algorithm is successfully completed, the image captured by the drone at that moment is defined as the keystone.

The third part of the crack detection process is that when the keystone is obtained after utilizing the crack information-assisted drone flight automatic control algorithm, the same lightweight segmentation model is used to process the low-resolution image and determine the location of the crack in the image. The original high-resolution image is then divided into several image patches with fixed sizes using the patch-based method (Lai, 2015). The patches containing cracks are selected using the results of the previous lightweight crack segmentation algorithm. After that, a high precision segmentation model with a large number of parameters and higher precision is used to segment cracks in the selected image patches with higher accuracy. Then, all the image patches are combined into a crack prediction mask with the same size as the original image. Finally, the proposed crack geometry information extraction algorithm is used to obtain accurate geometric information about the cracks. It should be emphasized that this part is also completed quickly on the onboard computer of the drone, but since it will not affect the flight trajectory of the drone, it does not need to be completed in milliseconds.

Since the location information of the drone and the distance between the crack and the drone have been obtained in the previous process, the spatial location distribution of the cracks can be easily obtained. The visualization of the overall crack detection process is illustrated in Figure 1. In this study, the necessary hardware device was a drone equipped with an onboard computer and a binocular camera.

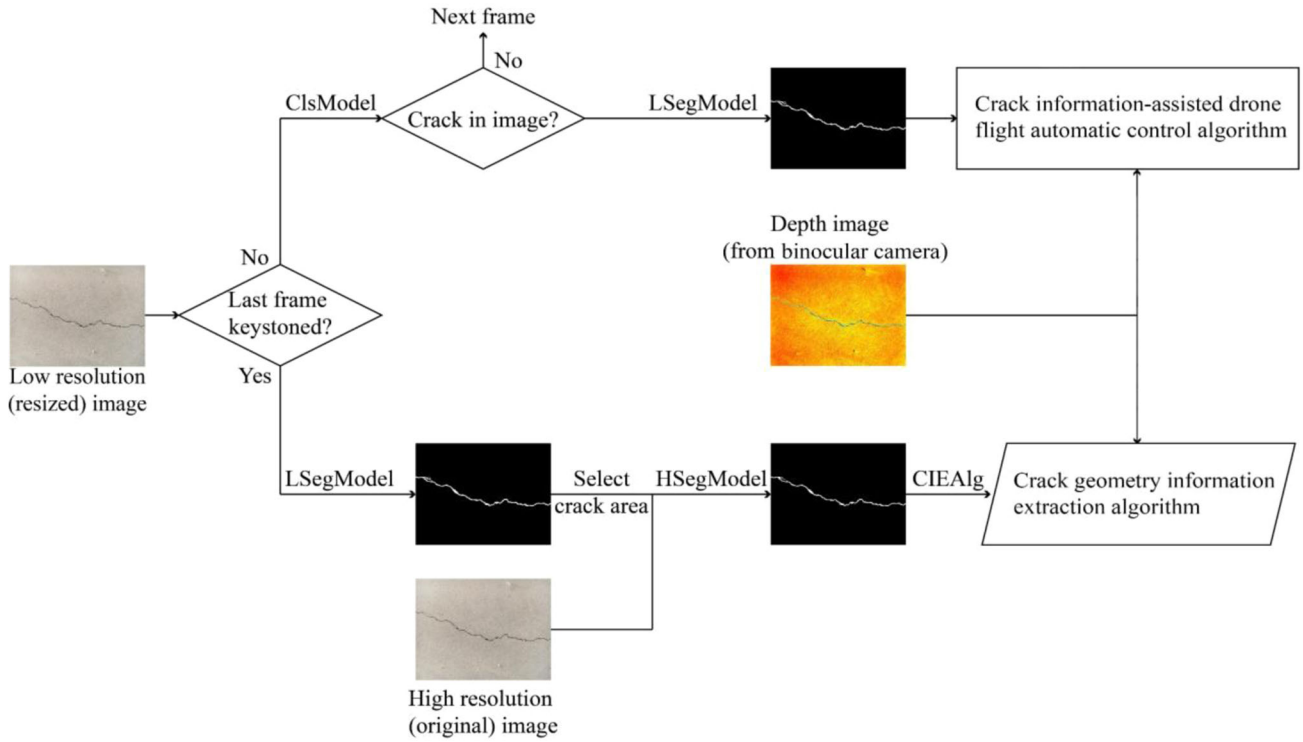

FIGURE 1 The flow chart of the whole process of crack detection.

2.2 | Lightweight crack classification algorithm

This paper focuses on the following points when choosing a lightweight crack classification algorithm. First, the selected lightweight model needs to have a very high inference speed, and second, it needs a high detection accuracy because, in this paper, it is necessary to guide the flight trajectory of the drone through the detection results. Therefore, if the inference speed of the model is greater than the speed at which the binocular camera captures images, it is guaranteed that every captured image has been processed, which can significantly improve the robustness of the method proposed in this paper. Second, the model's classification accuracy in the crack detection task should be as high as possible while ensuring speed. Therefore, to better implement the method proposed in this paper, ShuffleNet V2 (Ma et al., 2018) is selected as the lightweight crack classification model.

The detailed structure of the model is presented in Figure 2. The size of the input image of the model is $1024 \times 1024 \times 3$. The first layer of the model is a convolutional layer, which can increase the channel dimension to 24, followed by a max-pooling layer. After that, the model uses 16 basic blocks to process the feature map. The structure of the basic block is shown in Figure 3, where Figure 3a is an ordinary basic block, and Figure 3b is a basic block for downsampling. The basic blocks for downsampling are the first, fifth, and 13th steps. Their input channels are 24, 116, and 232, and their output channels are 116, 232, and 464. Next, a convolutional layer increases the number of channels of the feature map to 1024, and the global average pooling layer is used to convert the size of each channel to 1×1. At the end of the model, the feature map will be flattened and then input to the linear layer. The size of the output result is 2×1, which indicates whether the image contains cracks or not.

The detailed structure of the basic block is as follows. When downsampling is not required, the input feature map is subjected to the channel split operation, which splits the feature maps into two groups channel-wise and performs different operations as shown in Figure 3b. Here, Conv stands for the convolution operation, and DWConv for the depth-wise convolution operation. Finally, the two sets of processed feature maps are spliced in the channel dimension, and the channel shuffle operation is performed, which means the original order of the feature maps is shuffled in the channel dimension. When

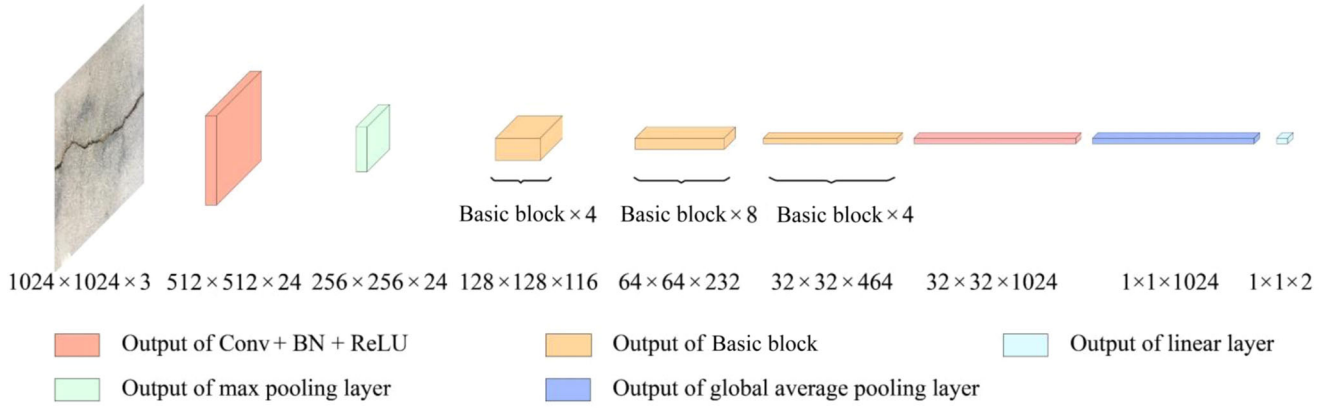

FIGURE 2 The overall network architecture of ShuffleNet V2. Conv represents the convolutional layer, BN means batchnormalization and ReLU indicates rectified linear units.

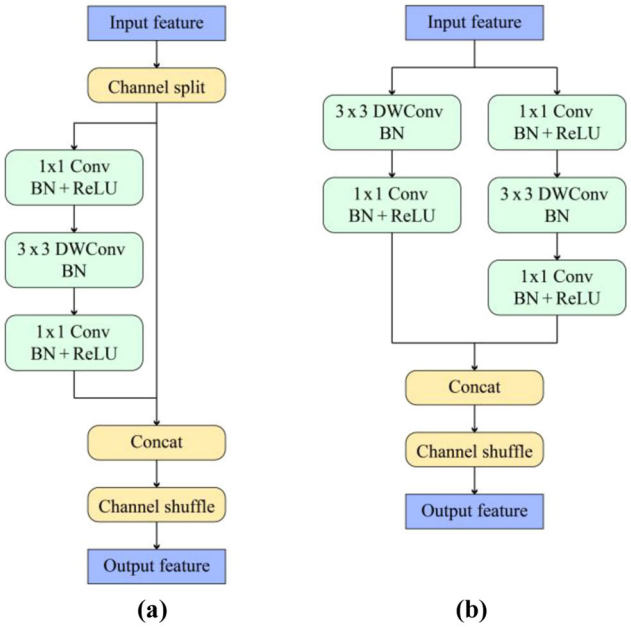

FIGURE 3 Architecture of modules in basic block: (a) shows ordinary basic block, and (b) is basic block for downsampling. DWConv represents the depth-wise convolution operation, Conv represents the convolutional layer, BN means batchnormalization and ReLU indicates rectified linear units.

downsampling is required, the feature maps are directly input to the two branches. After splicing, a channel shuffle operation is also performed.

2.3 | Lightweight crack segmentation algorithm

The principle of the model selected in this paper's lightweight crack segmentation algorithm is similar to that of the lightweight crack classification algorithm, and its inference speed is the most important rather than detection accuracy and parameter quantity. The detection effect of the lightweight crack segmentation algorithm does not affect the final crack detection result because the high-precision crack segmentation algorithm realizes the final geometric information extraction of the crack. The lightweight segmentation algorithm used in this paper is the bilateral segmentation network (BiSeNet V2; Yu et al., 2020), whose general network structure is shown in Figure 4. BiSeNet V2 separates spatial details and classification semantics to achieve highly accurate and highly efficient real-time semantic segmentation. It includes two branches: the detail branch and the semantic branch. The detail branch, with wide channels and shallow layers, captures low-level details and generates high-resolution feature representations. The semantic branch, which has narrow channels and deep levels, is used to obtain high-level semantic context. The semantic branch is lightweight because it reduces the channel capacity and uses a fast downsampling strategy. Behind the two branches, a bilateral guided aggregation layer connects and merges the two types of feature maps. Moreover, a booster training strategy has been developed, which can improve the segmentation performance without increasing the inference cost.

In the detail branch, the network is built according to the principle of VGGNet stacking (Simonyan & Zisserman, 2014), which contains six convolutional layers, batch normalization layers, and rectified linear units (ReLU) layers. The stride of the first, third, and fifth convolutional layers is two. In the semantic branch, any lightweight convolutional model can be used. In this paper, stem block, gather-and-expansion layer and context embedding block are used. The detailed structures are shown in Figure 5.

The use of global average pooling in the context embedding block of the semantic branch can effectively embed global context information.

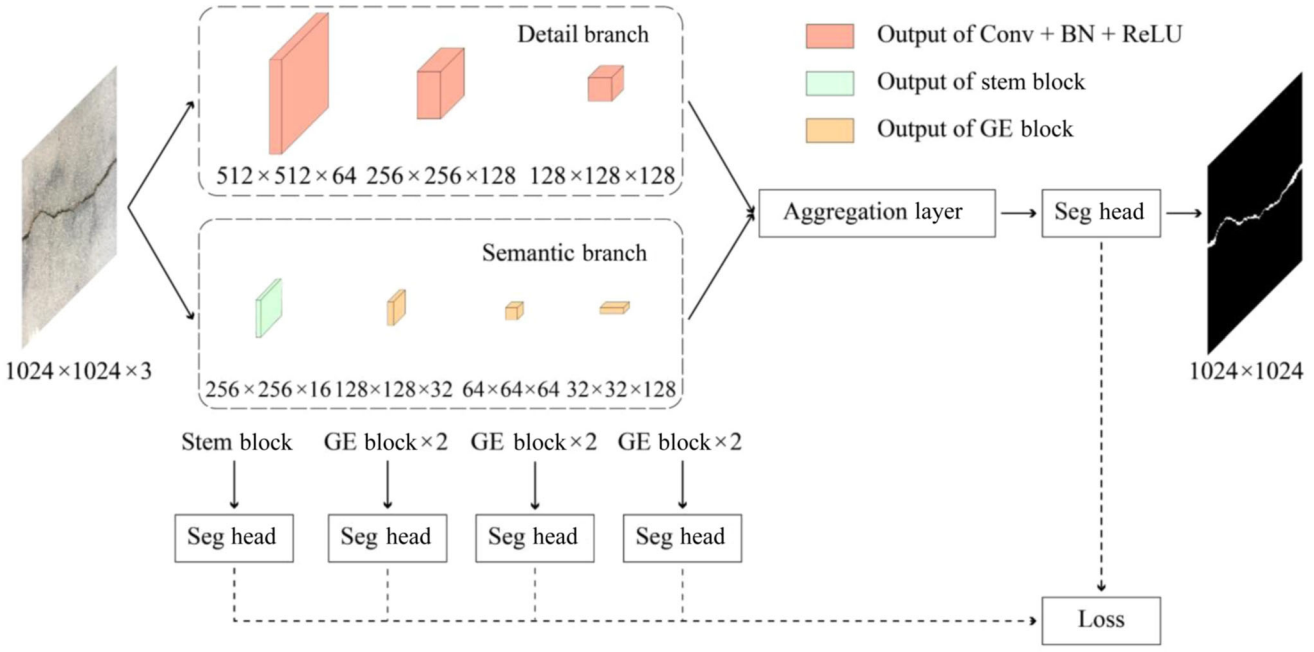

FIGURE 4 The overall network architecture of BiSeNet V2. GE Block means gather-and-expansion block, Conv represents the convolutional layer, BN means batchnormalization and ReLU indicates rectified linear units.

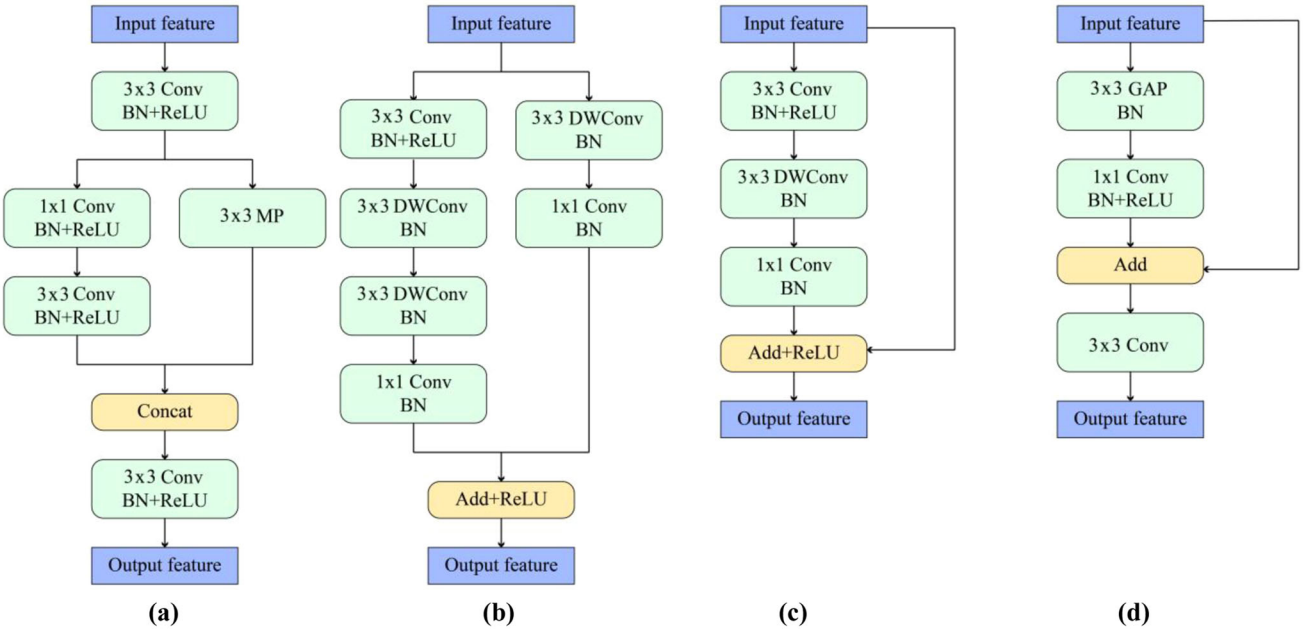

FIGURE 5 Detail structure of module in detail branch: (a) is stem block, (b) shows gather-and-expansion layer with downsampling, (c) represents ordinary gather-and-expansion layer, and (d) is context embedding block. GAP means global average pooling, MP means max pooling layer, Conv represents the convolutional layer, BN means batchnormalization, DWConv represents the depth-wise convolution operation and ReLU indicates rectified linear units.

A bilaterally guided aggregation layer is employed in the aggregation layer to aggregate the complementary information from the two branches as shown in Figure 6a. After the aggregation layer, use the Seg Head module to process the feature map and obtain the prediction result. The structure of the Seg Head module is shown in Figure 6b.

A booster training strategy is used in the process of model training. It can enhance the feature representation in the training phase and can be discarded in the inference phase. During the training process phase, the feature maps output from the first four blocks of the semantic branch are processed by the Seg Head module, and the four output

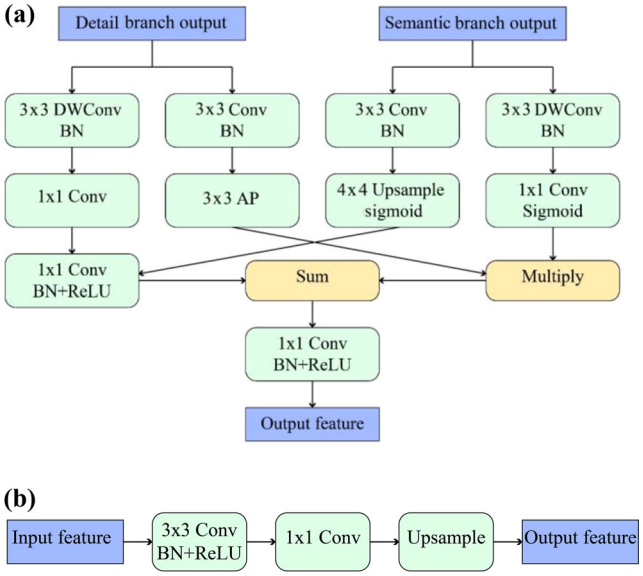

FIGURE 6 Detail structure of module for post-process two branches: (a) is bilaterally guided aggregation layer, and (b) represents Seg Head. AP means average pooling layer, Conv represents the convolutional layer, BN means batchnormalization, DWConv represents the depth-wise convolution operation and ReLU indicates rectified linear units.

FIGURE 7 The overall network architecture of the SegAttention Unet. Conv1x1 represents a convolution layer with a kernel size of 1×1, Conv3×3 represents a convolution layer with a kernel size of 3×3, BN means batchnormalization and ReLU indicates rectified linear units.

results are combined with the ground truth to calculate the loss and perform backpropagation.

2.4 | Crack information-assisted drone flight automatic control algorithm

The algorithm proposed in this paper, which is to automatically control the drone using crack information, must first roughly determine the flight path for the drone by manual or fixed-point navigation. Rough planning of the drone's route does not need to be detailed and can maintain a long distance from the building, so it is easy to implement. It is worth noting that the "long distance" in this paper refers to the minimum safe flight distance between the drone and the building, which depends on the type of drone and environmental factors. When the drone maintains a minimum safe flight distance from the building, it can ensure the drone's safety and discover as many millimeter-level cracks as possible. When the drone detects a crack on the flight path, it automatically approaches the surface where the crack is located to measure the crack width more accurately.

Sections 2.2 and 2.3 introduce the two lightweight crack detection algorithms. The lightweight crack detection algorithm can determine whether there are any cracks in the images captured by the camera. If the image contains cracks, the lightweight crack segmentation algorithm is used to perform fast segmentation to get the location of the crack in the image.

In this paper, a right-handed coordinate system is used, the y-axis is vertically upward of the camera, and the z-axis points to the front of the drone. The definition of the coordinate system is shown in detail in Figure 7. The coordinates of the crack in the drone camera coordinate system can be calculated from the two images captured by the binocular camera, which will be introduced in detail in Section 2.6. However, since the camera is inclined relative to the drone, and the camera is not necessarily parallel to the building when shooting, the result needs to be converted to the drone coordinate system to provide the location information of the crack. The pose of the camera can be obtained through the gyroscope. Thus, assume that the angle of rotation of the camera around the x-axis is α, and the angle of rotation of the camera around the y-axis is β. The coordinates of the crack pixel in the camera coordinate system are x_c, y_c, and z_c. Then through the coordinate system transformation, the coordinate calculation method of the crack pixel in the drone coordinate system is shown in Equation (1).

$$\begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} \quad (1)$$

where x_w, y_w, and z_w denote the coordinates of the crack pixel in the drone coordinate system.

In this paper, we define the moving target vector $\boldsymbol{T} \in \mathbb{R}^{3\times1}$ formed by where the camera points to the crack and

the adjusted moving target vector $\boldsymbol{D} \in \mathbb{R}^{3\times1}$. Furthermore, to prevent the drone from colliding with the building, we define the shortest distance d_t between the building and the drone. Take the average of the coordinates of all the crack pixels in the image in the drone coordinate system to obtain the moving target vector $\boldsymbol{T}'$ as shown in Equation (2). However, the drone needs to keep a certain distance from the shooting surface to prevent a collision. Therefore, it is necessary to modify the moving target vector $\boldsymbol{T}$, as shown in Equations (3) and (4). Since the drone cannot get too close to the building to prevent a collision, setting the shortest distance d_t between the drone and the building is necessary. Therefore, the target shortest distance d_t needs to be added to Equation (3). The value of d_t depends on the type of the drone and can be flexibly adjusted according to the working condition.

$$\boldsymbol{T}' = \frac{1}{N}\left(\sum_{i=1}^{N} x_{w,i}, \sum_{i=1}^{N} y_{w,i}, \sum_{i=1}^{N} z_{w,i}\right) \tag{2}$$

$$\boldsymbol{T} = \frac{R - d_t}{R}\left(T'_x, T'_y, T'_z\right) \tag{3}$$

$$R = \sqrt{T'^2_x + T'^2_y + T'^2_z} \tag{4}$$

where R denotes the calculated distance between the drone and the crack. d_t denotes the target shortest distance between the drone and the crack.

Flight control of drones can be achieved by using position controllers and speed controllers combined with real-time measurements of position and speed (Salih et al., 2010). However, since the drone will use the newly captured images to continuously calculate the position of the new target location during the movement, it is expected that the moving target vector $\boldsymbol{T}$ will be continuously updated. Besides, the accuracy of the calculation of the moving target vector $\boldsymbol{T}$ depends on the accuracy of the binocular camera to calculate the depth map, but the error of the calculation result using a single video frame is relatively large. Therefore, this paper proposes an algorithm that combines the inertial measurement unit (IMU) data information and the depth information calculated by the binocular camera to reduce the measurement error, which can ensure the stability of the flight. The crack information-assisted drone flight automatic control algorithm is shown in Algorithm 1. The following is a detailed explanation of the whole process and parameters in the algorithm and how to combine the detection results of the binocular camera with the IMU.

In Algorithm 1 of this paper, the keyframe is defined as the image that successfully calculates the crack position through the image of the binocular camera combined with the crack detection algorithms. $i_{\max}$ represents a very large number. At the beginning of the process, the moving target vector $\boldsymbol{T}$ needs to be calculated. Then set the drone's position P_1 at this time as the starting point. The role of P_1 is to facilitate the drone to return to the original position after the crack information-assisted drone flight automatic control algorithm is completed, so its positioning accuracy is not an essential factor.

ALGORITHM 1 Pseudocode for the crack information-assisted drone flight automatic control algorithm.

```
set c_in and c_out to 0
compute the original location P_1, T_1
for i = 2:i_max do
  if this frame is a keyframe then
    get the latest T_i
    compute I_{i-1}
    adjust n and compute D_i
    if|D_i| < Th_1 then
      c_in = c_in + 1
    else
      ifc_in ≠ 0 then
        c_out = c_out + 1
      end if
    end if
    if c_out > c_outmax then
      set c_in and c_out to 0
    end if
    if c_in > c_inmax then
      terminate the algorithm and obtain images at this moment
    for final crack detection break
    end if
    adjust the flight direction of the drone to the adjusted moving
    target vector D_i and move
  end if
end for
```

Before the algorithm terminates, the loop performs the following operations: Calculate the distance between the drone's position P_{i-1} in the previous keyframe and the current position P_i, namely, $\boldsymbol{I}_{i-1}$, through the IMU data. This paper defines $I \in \mathbb{R}^{3\times1}$. The calculated $\boldsymbol{I}_{i-1}$ is the displacement vector of the drone between two keyframes calculated by the IMU. The acceleration data measured by the IMU are integrated to obtain the displacement information of the drone between the two keyframes. Due to the short time interval between two keyframes and the high sampling frequency of the IMU, the displacement information obtained by the sensor has high accuracy.

Then, calculate the moving target vector $\boldsymbol{T}_i$ at this time calculated through the latest images of the binocular camera. Because the depth map calculated by the binocular

camera has a relatively large error, the moving target vector $\boldsymbol{T_i}$ needs to be adjusted when controlling the moving direction of the drone. In this paper, the error reduction is achieved by fusing the measurement results of n keyframes by sliding the time window. The calculation formula of the drone's adjusted moving target vector $\boldsymbol{D_i}$ is shown in Equation (5). Since the measurement error of the IMU is smaller than the error of the depth map calculated by the binocular camera, the variance of $\boldsymbol{D_i}$ decreases as n increases. In addition, the greater the distance from the measured object, the greater the error of the depth map calculated by the binocular camera. Thus, the forgetting factor α is introduced to reduce the overall error of the position estimation.

$$\boldsymbol{D_i} = \frac{\boldsymbol{T_i} + \sum_{j=1}^{n-1} \alpha^j \left(\boldsymbol{T_{i-j}} - \sum_{k=1}^{j} \boldsymbol{I_{i-k}} \right)}{1 + \sum_{j=1}^{n-1} \alpha^j} \tag{5}$$

where n is the number of the selected keyframes, the value of n is smaller than i, and the specific value of n can be dynamically changed during the flight of the drone since i increases with time. α is a value between zero and one. The closer α is to one, the more $\boldsymbol{D_i}$ tends to rely on the calculation results of the IMU. Then control the drone to move in the direction of the vector $\boldsymbol{D_i}$, and repeat the above operations to move the drone toward the crack surface.

In order to ensure that the drone can stabilize at the target position, several parameters are set to ensure that the drone is within the range of the target position within a certain period. The meanings of c_{in} and c_{out} are the number of keyframes when the drone is at the target position and outside the target position. Since the drone will be affected by the environment and other factors when hovering, it will cause a slight drift. Thus, it is necessary to set a parameter Th_1 to determine whether the drone reaches the target position stably. If the norm of vector $\boldsymbol{D_i}$ is smaller than Th_1, increase c_{in} by 1; otherwise, if c_{in} is not 0, increase c_{out} by 1. The smaller the Th_1 is selected, the more stable the drone needs to be.

$c_{in\max}$ is the maximum number of keyframes at which the drone is at the target position. If c_{in} is greater than $c_{in\max}$ it is considered that the drone has been hovering stably at the target position, and the program is terminated. The meaning of $c_{out\max}$ is the maximum number of keyframes that the drone exceeds the target position. If the drone is not at the target position for a long time, the drone may not hover stably due to drift and other reasons. If c_{out} is greater than $c_{out\max}$, set c_{in} and c_{out} to 0. The values of c_{in} and c_{out} are both set to 0 at the beginning of the process. The above operations can be used to judge whether the drone has reached the vicinity of the target location and ensures the drone flights stability. The larger the selection of $c_{in\max}$ and $c_{out\max}$ is, the longer the drone will remain at the target position. In this case, the drone is less likely to shake, and the camera can capture a clearer image. Conversely, the time required for drone shooting will be longer. In addition, if the value of Th_1 is too small, it will be difficult for the drone to meet the requirements of the algorithm.

Finally, the previously recorded location information P_1 can be used to control the drone to return to P_1 through fixed-point navigation, and then continue to perform crack detection on other locations. In addition, since the drone may encounter obstacles during the flight, it is necessary to arrange a multi-directional binocular camera on the drone combined with an obstacle avoidance algorithm so that the drone is not prone to collision. Moreover, when the drone successfully approaches the building, the position of the drone in the world coordinate system can be obtained by positioning methods such as GPS. By combining the position of the drone and the distance calculation results of the binocular camera, the absolute position of the crack can be easily obtained.

2.5 | High-precision crack segmentation algorithm

When the drone reaches the neighborhood of the measured crack, a segmentation model with more parameters and higher accuracy must be used to achieve high-precision crack detection and obtain more accurate crack edges. Using the BiSeNet V2 presented in Section 2.3 with the downsampled image, the image patch containing the cracks can then be obtained from this crack edge in a subsequent step. In this paper, the resolution of the original image refers to the resolution of the image captured by the binocular camera. Thus, the resolution of the original image changes depending on the choice of the binocular camera. In order to achieve higher-precision crack detection, it is necessary to use a relatively high-resolution binocular camera. Due to the high image resolution, directly inputting the original image into a semantic segmentation network requires excessive graphic processing unit (GPU) memory and computing power. Therefore, the original image is first sliced using a sliding window of 1024 × 1024 pixels, where there is an overlap of 512 pixels between every two adjacent windows. Upsample the output result of the previous BiSeNet V2 model to the original resolution and calculate the number of crack pixels contained in each sliced image patch. If the number of pixels representing cracks exceeds the set threshold, this one image patch is input to the high-precision semantic segmentation model. Finally, a merge of all output results

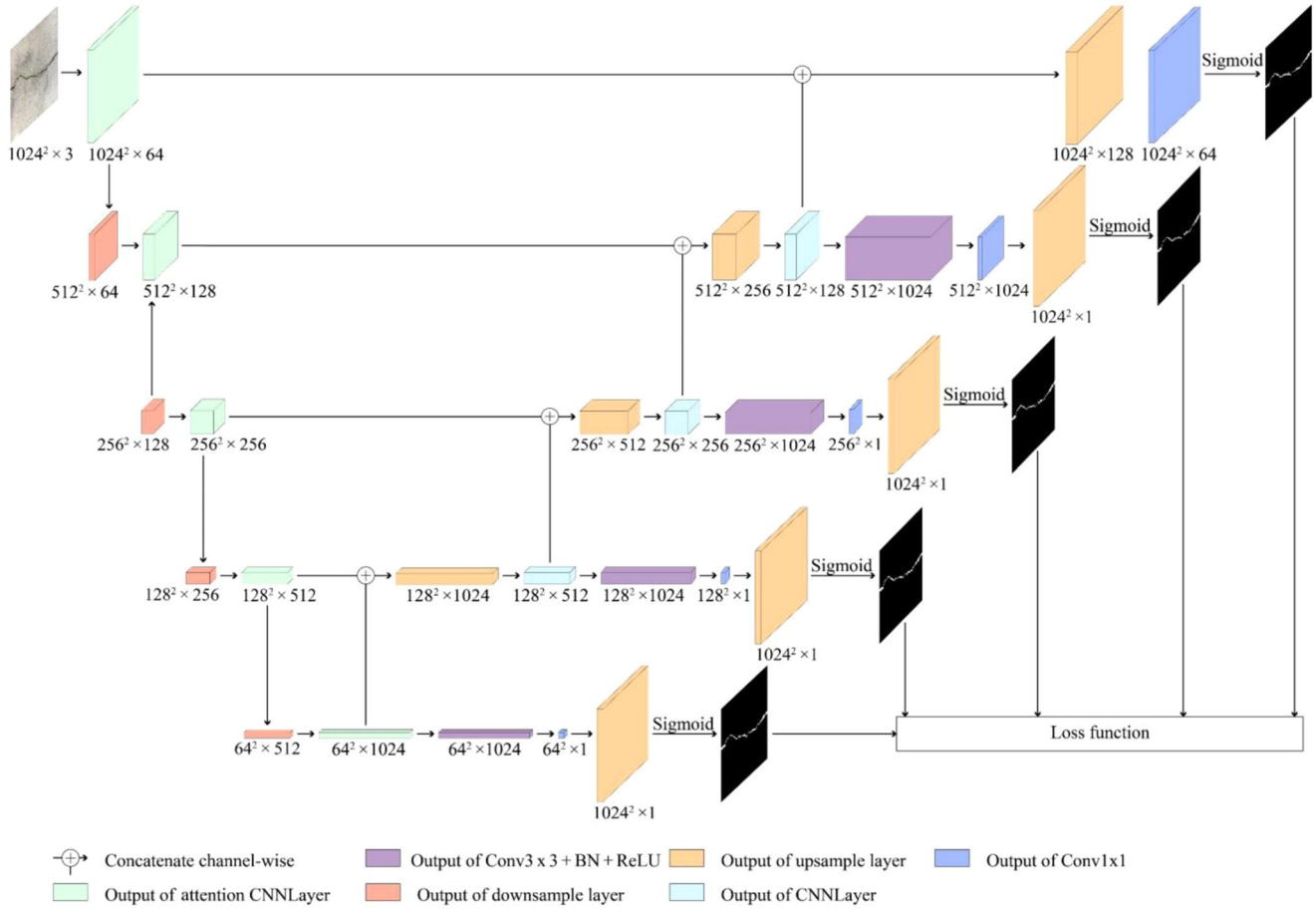

FIGURE 8 Architecture of SegAttention UNet main component: (a) is spatial attention module, and (b) is channel attention module. MLP represents multilayer perceptron, GMP means global max pooling layer, GAP means global average pooling layer, AP represents average pooling layer, MP represents max pooling layer, Conv indicates the convolutional layer, BN means batchnormalization and ReLU indicates rectified linear units.

is obtained, and they are stitched back to their original position. Since the segmentation result of the edge part of the patch by the semantic segmentation model is poor, only the middle part of the sub-image is selected for splicing. For an $n \times n$ patch that is not at the edge of the original image, extract an $n/2 \times n/2$ sub-image, in which the center is the same as the original patch. For the edge part of the original image, the segmentation result of the patch is directly used, and the overlapping part takes the average value of multiple patches. This processing method effectively avoids the poor effect of semantic segmentation on image edges. After the above operations, a binarized mask with the same size as the original image is obtained.

Because the high-precision fracture segmentation algorithm does not need to have extremely fast speed but requires extremely high precision, therefore, based on the traditional UNet (Ronneberger et al., 2015), this paper proposes a SegAttention UNet. First, to improve the ability of model feature extraction, SegAttention UNet adds two self-attention mechanism modules during downsampling, including the spatial attention and channel attention module. Furthermore, higher computational accuracy is achieved by increasing the number of channels. Finally, to improve the effect of model training, a branch is added after each tensor before upsampling. In this branch, the crack segmentation result is obtained through the convolutional layer and upsample calculation and participates in the loss calculation. This approach can help the convolutional layers inside the network achieve more accurate feature extraction. The network structure is shown in Figure 8.

In terms of the details of the network structure, SegAttention UNet downsamples the input crack image four times for feature extraction and then decodes it through deconvolution. By adding the tensors before downsampling and the tensors after deconvolution in the channel dimension, different levels of features can be combined. CNNLayer comprises two convolutional layers, two batch norm layers, two dropout layers, and two Leaky ReLU layers in turn. The module used for each downsampling is

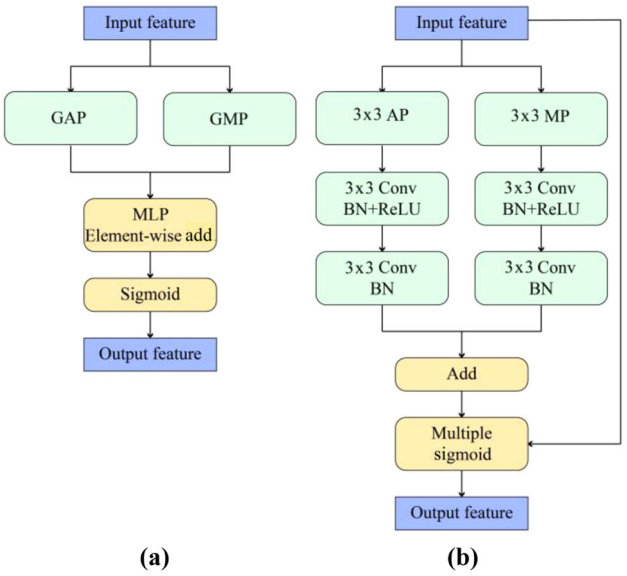

FIGURE 9 The visualization of the detection result of the high-precision segmentation algorithm. The images shown in the left half of the figure are the detection result of the SBGCrack dataset, and the images shown in the right half of the figure are the detection result of the METUCrack dataset.

attention CNNLayer, which adds spatial attention module and channel attention module in turn before CNNLayer. At the end of the SegAttention UNet, feature mapping is performed through the convolution layers and sigmoid layers to obtain a binary image of the crack segmentation result. To improve the performance of the SegAttention UNet, new branches are added to the four parts before the upsampling module as shown in Figure 8. These branches map four tensors of size $64 \times 64 \times 2048$, $128 \times 128 \times 1024$, $256 \times 256 \times 512$, and $512 \times 512 \times 256$ into 1024 channels. Afterward, these four tensors are mapped into tensors with a channel number of 1 through convolutional layers and sigmoid layers. During model training, the final output of the model and the above four tensors are added after calculating the loss values. When the model performs crack detection, only the results of the last layer in the model need to be output.

It should be noted that it is acceptable to use a model with a large number of parameters and a slower inference speed since this process does not need to be completed in real time. In addition, any other high-precision semantic segmentation model can be used to replace the SegAttention UNet (Figure 9).

2.6 | Crack width measurement

The crack detection process described above can be used to determine the pixel set in which the crack is located in the three-channel color image. Using the binarized segmentation results, this paper proposes a maximum crack width measurement algorithm to calculate the maximum crack width. Since there may be multiple cracks in each image, it is necessary to use the connected domain labeling algorithm to extract each crack separately to obtain an image set C, in which each image C_i in the set C contains only one crack. Then, the maximum crack width is calculated separately for each image C_i. The specific calculation method is as follows. First, all crack pixels will be traversed. If not all the pixels in the four neighborhood pixels of a pixel are crack pixels, this pixel will be added to the set B. The four neighborhood pixels $N_4(p)$ refers to the four pixels adjacent to a pixel p in the up, down, left, and right directions, and the set B is the set of edges of cracks. Then, the image needs to be processed by the skeleton line extraction algorithm to obtain the crack skeleton line S, which can be realized by the Zhang–Suen algorithm (T. Y. Zhang & Suen, 1984). For each point S_j in S, calculate the minimum distance $w_{\min,i}$ from all elements in the set B. Among them, the distance $dist(B_i, S_j)$ is the Euclidean distance. Finally, take the maximum value of all $w_{\min,i}$ and multiply by 2 to get the maximum crack width W. The detailed expression of the above calculation method is shown in Algorithm 2. Where $N_4(p) \neq 1$ means that the four neighborhood pixels of pixel p are not all crack pixels. M denotes a large value for initializing $w_{\min,i}$.

However, the maximum crack width obtained by Algorithm 2 is only the crack width in the image coordinate system and not the crack length in the world coordinate

ALGORITHM 2 Pseudocode for the maximum crack width measurement algorithm.

process image using connected component labeling algorithm
for each $C_i \in C$ **do**
  for each $p \in C_i$ **do**
    if $N_4(p) \neq 1$ **then**
      add p into B
    end if
  end for
  set $w_{\max} = 0$
  for each $S_i \in S$ **do**
    set $w_{\min,i} = M$
    for each $B_j \in B$ do
      $w_{\min,i} = \min(dist(S_i, B_j), w_{\min,i})$
    end for
    $w_{\max} = \max(w_{\min,i}, w_{\max})$
  end for
  $W = 2w_{\max}$
end for

system. Therefore, it is necessary to calculate the pixel resolution η of the image.

This paper uses the depth map to calculate the pixel resolution η. The depth map can be obtained by using the binocular camera, and the coordinates of the pixel in the camera coordinate can be inversely calculated through the depth map. It should be noted that converting the world coordinate system to the local coordinate system with the camera as the origin is the camera coordinate system.

In this paper, the depth map calculation needs to be calculated from the image of the binocular camera, which can also be used as the critical decision information for calculating the moving target vector T of the crack information-assisted drone flight automatic control algorithm in Section 2.4. The following operations are required: camera calibration, epipolar rectification, stereo matching, and depth map calculation. First, the camera calibration needs to calibrate the left and right cameras to obtain the camera intrinsic parameters, distortion coefficients, and camera extrinsic parameters. The camera extrinsic parameters refer to the rotation matrix and the translation matrix between the left and right cameras (Z. Zhang, 2000). The operation of obtaining the undistorted image is as follows: first, the pixel coordinate system of the obtained image is converted to the camera coordinate system through the internal parameter matrix, and then the distortion correction is performed in the camera coordinate system. After that, the camera coordinate system is re-transformed to the image pixel coordinate system and interpolates the pixels of the new image with the pixel values of the original image. In the epipolar rectification part, the purpose is to greatly improve the efficiency of the stereo matching search in the subsequent steps. After epipolar rectification, the subsequent stereo matching only needs to search the same height in the two images. Epipolar correction can be achieved by the Bouguet epipolar rectification method (Bouguet & Perona, 1998). In terms of stereo matching, the purpose is to find its corresponding point in the right image for each pixel in the left image, which is the same point in the world coordinate system, so that the disparity can be calculated. The calculation process of the stereo matching algorithm needs to go through the following stages: matching cost calculation, cost aggregation, parallax optimization, and parallax refinement, which can be achieved through the semi-global block matching (SGBM) algorithm (Hirschmuller, 2008). After the disparity map is generated by stereo matching, the disparity map needs to be filtered and processed subsequently to improve the quality of the disparity map. Disparity map filtering can convert sparse disparity into dense disparity, reduce disparity map noise to a certain extent, and improve the visual effect of the disparity map. The filtering of the disparity map can be realized by the weighted least squares filtering algorithm (Farbman et al., 2008). Finally, the calculation of the depth map can be realized by the disparity map, and the calculation of the coordinates of each pixel in the camera coordinate system can be realized by using the previously calculated disparity through Equation (6). In this paper, a right-handed coordinate system is used, the y-axis is vertically upward of the camera, and the z-axis points to the front of the camera.

$$\begin{cases} x_c = \frac{(p_x - c_x)T_x}{-d + c_x - c'_x} \\ y_c = \frac{(p_y - c_y)T_x}{-d + c_x - c'_x} \\ z_c = \frac{T_x}{-d + c_x - c'_x} \end{cases} \tag{6}$$

where x_c, y_c, and z_c are the coordinates of the pixel in the x, y, and z axis in the camera coordinate system, respectively. p_x, p_y denotes the x and y coordinates of the pixel in the image coordinate system, respectively. c_x and c_y are the offsets of the optical axis in the image coordinate system in the left eye camera, which are included in the camera intrinsic parameters. c'_x is the offsets of the optical axis in the x-direction of the right camera. d is the disparity of the two points calculated by the SGBM algorithm. f is the focal length. T_x is the translation between the projection centers of the two cameras and is a negative value.

In order to reduce the calculation error of the pixel resolution of the crack area, since the area of the image occupied by the crack is generally small, the average pixel resolution of all pixels in the detected crack area is selected as the pixel resolution of a single crack. First, the calculation of the average pixel resolution requires the use of the image set C extracted using Algorithm 2, and the calculation of the average pixel resolution is performed separately for each crack. Through the crack edge extraction method in Algorithm 2, the crack edge image composed of set B can be obtained, and the diameter of the edge is 1 pixel. After that, traverse the crack edge image row by row and column by column. If there are two crack edge pixels in a row or column, they are used as a value in Equation (7) to participate in the calculation and finally get the average pixel resolution of each crack. The calculation method of average pixel resolution is shown in Equation (7).

$$\eta = \frac{1}{M+N}\left(\sum_{i=1}^{N}\left|\frac{x_{1,i} - x_{2,i}}{p_{x1,i} - p_{x2,i}}\right| + \sum_{i=1}^{N}\left|\frac{y_{1,i} - y_{2,i}}{p_{y1,i} - p_{y2,i}}\right|\right) \tag{7}$$

where N represents the total number of rows of two crack edge pixels in a row in the crack edge image, and correspondingly M represents the total number of columns

that contains two crack edge pixels. $x_{1,i}$ and $x_{2,i}$ are the coordinates of two crack edge pixels in a row in the world coordinate system, and $y_{1,i}$ and $y_{2,i}$ are the coordinates of two crack edge pixels in a column in the world coordinate system. $p_{x1,i}$ and $p_{x2,i}$ are the coordinates of the two crack edge pixels in a row in the image coordinate system, $p_{y1,i}$ and $p_{y2,i}$ are the coordinates of the two crack pixels in a column in the image coordinate system. Finally, multiply the pixel resolution η and W of each crack in the image to get the maximum crack width in the world coordinate system.

3 | EXPERIMENTS

3.1 | Crack detection algorithms experiment

3.1.1 | Dataset description

To test the efficiency and generalization of the crack detection-related algorithms proposed in this paper, two different public datasets were selected. The images of the first dataset (METUCrack) were captured in different buildings located at Middle East Technical University (Özgenel, 2019). The whole dataset includes 458 images of concrete surface cracks. The size of each image is 3024×4032 pixels. The ground truth is the binarized image labeled with a set of cracked pixels. The second dataset (SBGCrack) is from Project 1 of the First International Project Competitions of Structural Health Monitoring (Bao et al., 2021). The images include 200 original fatigue cracks images with a resolution of 4928×3264, all captured by different bridge inspectors with various camera parameters in the steel box girder. The images in these two datasets are scaled to 1024×1024 pixels to match the situation applied in the actual detection. Then, split each dataset into training, validation, and test sets in a ratio of approximately 7:2:1.

It should be noted that the training of all deep learning models in all experiments in this section was done on a workstation using four Nvidia Tesla V100s. However, the inference of the model is finished on the Nvidia Jetson AGX Xavier using the Linux operating system, so the statistic results such as mean intersection over union (MIoU) and frames per second (FPS) are also the test results on the onboard computer. In addition, to improve the inference speed of the model, the TensorRT library was used to convert the PyTorch model into a TensorRT model, thereby achieving higher FPS. All crack detection algorithms in Sections 3.1.2, 3.1.3, and 3.1.4 are built, trained, and tested through the PyTorch framework.

3.1.2 | Experiment of crack classification algorithm

The crack classification algorithm uses three effect models indexes and two lightweight models indexes: accuracy, precision, recall, reasoning speed and parameter quantity. The following hyperparameter values were used: learning rate = 0.00001, optimizer = Adaptive Moment Estimation (Adam) (Kingma & Ba, 2014), batch size = 128, momentum = 0.9, weight decay factor = 0.002. A total of 100 epochs have been learned. The cross-entropy loss was chosen to calculate the model loss.

Table 1 compares the ShuffleNet V2 with the commonly used crack classification algorithms, including VGG-19 (Simonyan & Zisserman, 2014), ResNet50 (He et al., 2016), SE-ResNet (Hu et al., 2018), convolutional block attention module (CBAM)-ResNet50 (Woo et al., 2018), and another well-known lightweight classification algorithm MobileNet V2 (Sandler et al., 2018). It can be seen that ShuffleNet V2 loses only 0.32% of accuracy, compared to CBAM-ResNet50 with the highest accuracy in METU-Crack and has the highest accuracy in SBGCrack. The reasoning speed of ShuffleNet V2 is more than three times that of the traditional crack classification algorithm, and the model size is only 1/20 of CBAM-ResNet50. This considerably simplifies the deployment and operation of edge computing devices while also meeting the real-time detection requirements of drones.

3.1.3 | Experiment of lightweight crack segmentation algorithm

In the experiment for the lightweight crack segmentation algorithm, accuracy, precision, recall, and MIoU are used to evaluate the accuracy of the model. The reasoning speed and model parameters are used to evaluate the lightweight degree of the model. The loss function of all models is designed as the sum of binary cross entropy (BCE) loss and soft dice loss. Other hyperparameters include learning rate = 0.00001, optimizer = Adam, batch size = 128, momentum = 0.9, and weight decay factor = 0.002. A total of 100 epochs have been learned.

Table 2 details the comparison of the BiSeNet V2 and other latest algorithms on two datasets, including DenseNet121-FCN (Li et al., 2019), SegNet (Badrinarayanan et al., 2017), DlinkNet50 (L. Zhou et al., 2018), and DeepLab-v3+ (L.-C. Chen et al., 2018). BiSeNet V2 is a lightweight segmentation model with clear advantages in terms of model size, and its inference performance is significantly faster than that of other segmentation models. The mIoU of Bisenet V2 (53.00/76.80) did not decrease

TABLE 1 Comparison with the popular crack classification algorithms

Method	Accuracy (%)	Recall (%)	Precision (%)	FPS	Parameters (M)	giga floating point operations (GFLOPs)
VGG-19	83.46/97.41	75.47/96.92	89.83/97.88	9.47	38.93	408.02
ResNet50	85.37/97.14	75.41/95.01	94.16/96.12	27.50	23.51	85.41
SE-ResNet	87.06/97.30	77.55/96.44	95.76/98.12	15.73	26.03	85.42
CBAM-ResNet50	87.37/97.01	83.83/95.42	90.22/98.56	12.06	26.23	85.45
MobileNet V2	86.13/97.55	74.40/97.11	97.21/97.98	58.92	2.23	6.26
ShuffleNet V2	87.02/97.65	80.22/98.47	92.84/96.87	78.79	1.26	3.01

Note: The left data represent result in METUCrack, and the right data represent result in SBGCrack. The test results in the table are represented by a/b, where a represents result in METUCrack, and b represents result in SBGCrack.

TABLE 2 Comparison with the popular crack segmentation algorithms

Method	Recall (%)	Precision (%)	MIoU (%)	FPS	Parameters (M)	GFLOPs
DenseNet121-FCN	65.48/92.32	52.02/76.45	40.24/71.68	16.62	7.82	70.08
SegNet	78.61/94.67	65.98/81.55	54.26/77.66	3.11	29.44	641.63
DlinkNet50	71.04/94.30	70.25/79.30	51.38/75.60	4.25	120.18	217.65
DeepLab-v3+	60.10/81.83	73.51/93.91	48.03/77.44	5.81	59.34	354.13
BiSeNet V2	78.42/94.65	63.10/80.59	53.00/76.80	32.87	3.34	48.90

Note: The left data represent result in METUCrack, and the right data represent result in SBGCrack. The test results in the table are represented by a/b, where a represents result in METUCrack, and b represents result in SBGCrack.

significantly, compared with DenseNet121-FCN (40.24/71.68), SegNet (54.26/77.66), DlinkNet50(51.38/75.60), and DeepLab-v3+ (48.03/77.44). As a result of the preliminary segmentation, sufficient data and information have been provided for the following high-precision segmentation.

3.1.4 | Experiment of the crack detection process

The lightweight crack segmentation and the high-precision crack segmentation algorithm are integrated into the process to detect the crack edge and compare it with the results of other processes. In the high-precision crack detection process, which is the third step of the whole process, BiSeNet V2 is used in the overall process with the same parameters as those mentioned in Section 3.1.3. SegAttention UNet is used as part of the high-precision segmentation algorithm, and its loss function is designed as the combination of BCE loss and soft dice loss. The relevant parameters during training are as follows: learning rate = 0.00001, optimizer = Adam, batch size = 128, momentum = 0.9, weight decay factor = 0.002. A total of 100 epochs have been trained.

The visual representation of the prediction results is shown in Figure 10. It should be noted that the results of the high-precision crack segmentation algorithm shown in this section are not just the results of a single prediction using the SegAttention UNet. This section uses the method mentioned in Section 2.1, which divides the original image into multiple patches, predicts them separately, and then performs data fusion. From the results, it can be seen that the model can still effectively identify the crack pixels despite some errors in the labels used for training. Furthermore, using the method proposed in this paper, the MIoU is improved on both datasets, compared to using other algorithms directly as shown in Tables 2 and 3. The effectiveness of the high-precision crack segmentation algorithm proposed in this paper has been demonstrated.

3.2 | Overall method experiment on the two-story building

3.2.1 | Drone system

Drones offer excellent maneuverability and a wide inspection range. They can move fast and capture images near

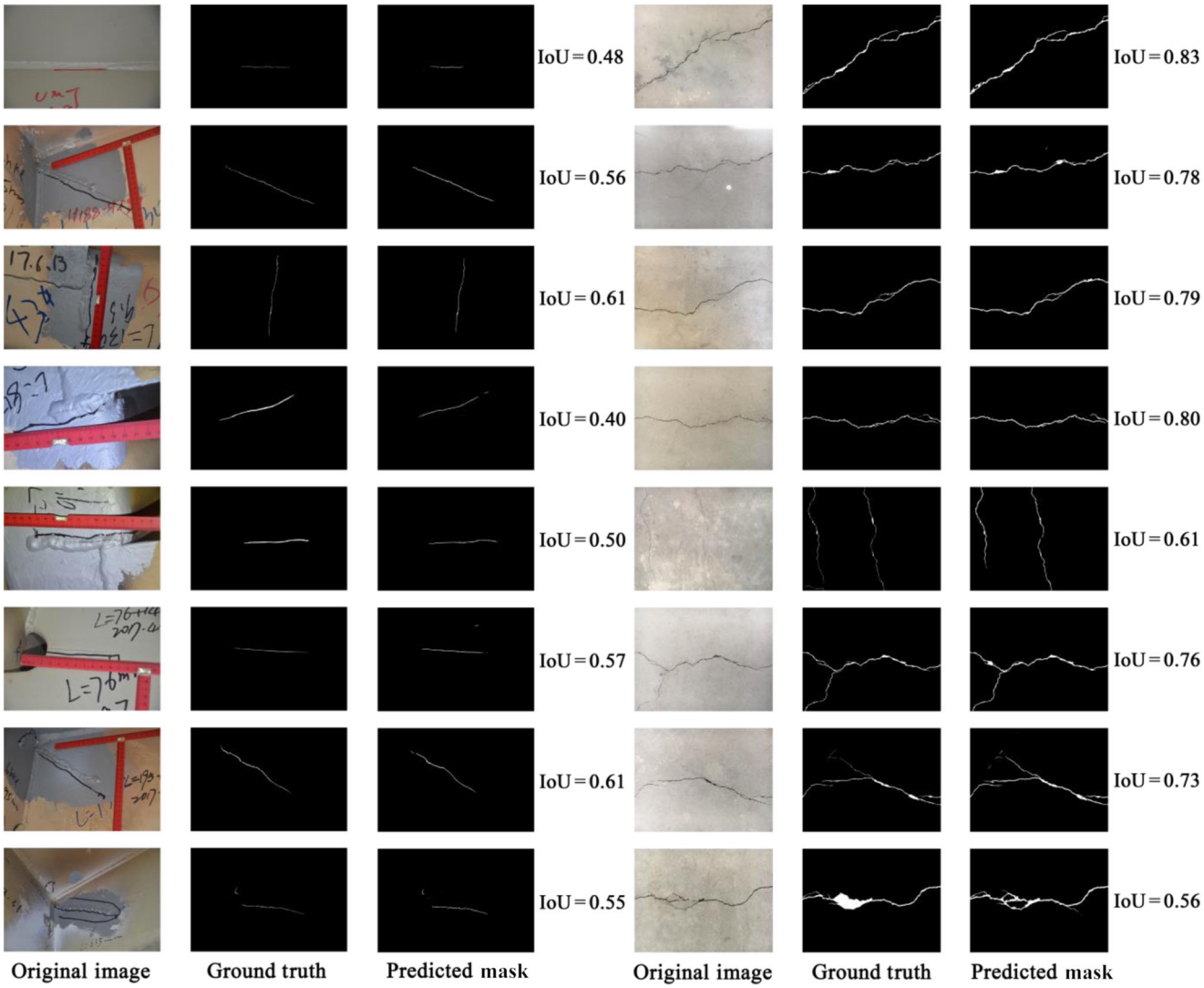

FIGURE 10 Components of the drone system and the definition of the coordinate system

TABLE 3 Comparison with the different crack detection process

Method	Recall (%)	Precision (%)	MIoU (%)
BiSeNet V2	78.42/94.65	63.10/80.59	53.00/76.80
BiSeNet V2 + SegAttention UNet	72.87/88.30	71.13/88.07	55.19/78.69

Note: The left data represent result in METUCrack, and the right data represent result in SBGCrack. The test results in the table are represented by a/b, where a represents result in METUCrack, and b represents result in SBGCrack.

various types of structures. Therefore, the deployment of the algorithm on the drone allows for wide-range damage identification of apparent structural cracks of the entire building structure. To achieve real-time detection of structural surface cracks using drones, it is necessary to carry a binocular camera to capture color information of the structural surface and obtain depth maps to detect crack geometry information. In addition, since the crack detection algorithm requires high computing power, an onboard computer needs to be integrated into the drone to process the images captured by the binocular camera online.

In this work, the DJI M300 RTK flight platform with dimensions of 810 × 670 × 430 mm, with a maximum weight of 2.7 kg was employed. Moreover, the drone applications can also be developed using drone development kits for applications such as on-board software development kit (SDK) (OSDK) and pay-load SDK (PSDK). The

TABLE 4 Key parameters of ZED2

Key parameters	Value
Focal length	2.2 mm
Image resolution	2K
Default FPS	15
Field angle of view	$110°(H) \times 70°(V) \times 120°(D)$

PSDK and OSDK are mainly used for secondary development. The role of PSDK is to implement functions such as gimbal control and camera control load coordination. OSDK is developed to perform functions such as flight control and obstacle avoidance of the drone.

The task of the onboard computer is to execute the PSDK and OSDK programs, collect data from the binocular camera, and process the online acquired image data online in real-time for crack detection. Due to the high computational complexity of the various algorithms in Section 2, the Jetson AGX Xavier is used as the onboard computer on the drone in the experiments. The Jetson AGX Xavier can achieve the performance of a GPU workstation on an embedded module while consuming less than 30 W of power, making it well-suited for edge computing. Since Jetson AGX Xavier uses NVIDIA GPUs, the TensorRT library can be used for high-performance inference acceleration of GPUs when performing model inference. The ZED2 Stereo Camera was selected as a binocular camera for image acquisition. The camera is equipped with an accelerometer, a magnetometer, and a gyroscope and therefore has great positioning capabilities. Other important parameters are listed in Table 4. The data acquisition frequency of the IMU is 400 Hz, so the accumulated error is relatively small. The displacement is obtained by integrating twice the acceleration data from the IMU. The onboard computer, Jetson AXG Xavier, is mounted on the upper part of the drone. A binocular camera is mounted on the lower part of the drone through the XPort gimbal. To achieve the pitch rotation adjustment of the binocular camera, the XPort gimbal is used to support the binocular camera. When the pitch rotation control is performed, the relative angle control amount, the absolute angle control amount, or the speed control amount must be converted into the speed to control the rotation of the gimbal according to the attitude and the rotation speed of the gimbal, and the gimbal rotation is controlled according to that speed. The overall structure of the drone, as well as the load, is shown in Figure 7.

In the drone experiment in this paper, drone flight control is realized by calling OSDK through C programming, and all crack detection algorithms are realized by python based on the PyTorch framework. Besides, the crack detection and drone flight control code are run parallel on Jetson AGX Xavier through multi-threading, and cross-language multi-threaded communication is realized through the Socket library.

3.2.2 | Experiment and result

The automatic control algorithm was developed through OSDK and deployed on the Jetson AGX Xavier. The experiment was conducted in front of a building at Tongji University.

In this experiment, the flight strategy was planned through fixed-point navigation. First, 18 control points are roughly set around the building, and the distance between each control point and the building surface is about 3 m. The OSDK is then used to control the drone to fly over each point individually to collect images on the building's surface. When the drone detects a crack in the image, the drone's position at that time is recorded, and the algorithm is used to automatically control the drone using crack information to guide the drone toward the crack. When the high-precision crack segmentation algorithm is completed, the drone returns to the previously recorded position and moves to the next control point. Due to the complicated environment in this experiment and the errors in the result of the lightweight crack detection algorithm, the crack information-assisted drone flight automatic control algorithm is only activated when more than five consecutive frames of images are identified as cracks. Furthermore, in this experiment, the values of c_{outmax}, c_{inmax}, Th_1, d_t and α are set to 10, 50, 0.5, 0.5, and 0.9, respectively. The value of the sliding time window changes dynamically. When the number of keyframes is less than 10, n is equal to the number of keyframes, and when the number of keyframes is greater than 10, n is equal to 10. The overall drone path is shown in Figure 11 to demonstrate the effectiveness of the crack information-assisted drone flight automatic control algorithm. In addition, the crack detection model used in the experiments is trained using the SBGCrack dataset and does not utilize the prior information in the experimental environment. Higher detection accuracy can be achieved if the model training is performed using the crack data collected in the experimental environment.

In order to verify the effectiveness of the crack information-assisted drone flight automatic control algorithm, the high-precision segmentation model is used to extract the edges of cracks in the two images before and after the drone approached the building surface. The visualization of the detection results is shown in Figure 12. The calculation results of the accuracy, recall, precision, and MIoU of crack edge extraction in the two cases are shown in Table 5.

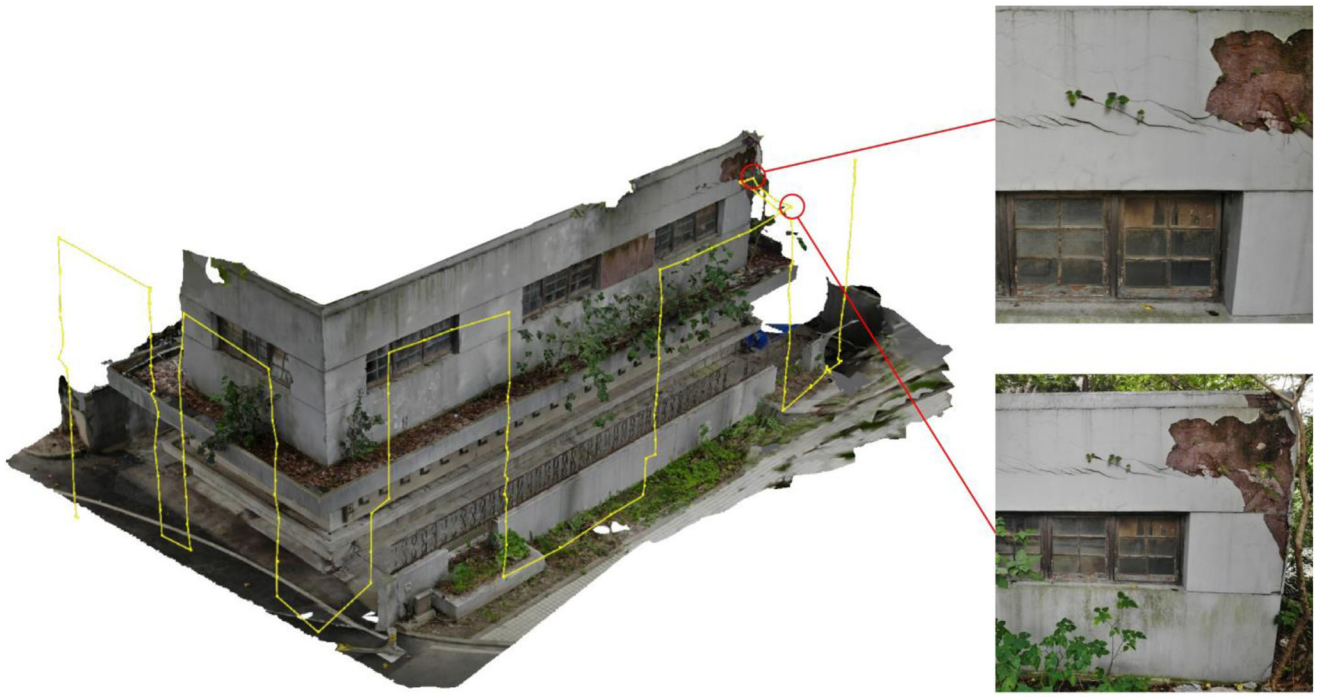

FIGURE 11 Experimental results of the automatic control algorithm. The yellow line in the figure shows the drone movement route. The red circle on the right side indicates the position of the drone before it detected the crack, and the red circle on the left side indicates the final stable position of the drone piloted by the automatic control algorithm. The two images show what the drone captured while in the corresponding positions.

TABLE 5 Comparison of detection results before and after using the crack information-assisted drone flight automatic control algorithm

Use drone flight automatic control algorithm	Recall (%)	Precision (%)	MIoU (%)
Yes	69.96	52.38	42.73
No	64.51	29.32	25.25

In order to verify the crack width measurement method proposed in this paper, the width measurement of millimeter-scale building surface cracks was carried out in this experiment. A total of 19 cracks were extracted without using the crack information-assisted drone flight automatic control algorithm, and the average error of crack width measurement was 2.84 mm. When using the algorithm, 22 cracks were extracted with an average error of 0.48 mm. In this experiment, there were 24 millimeter-scale cracks on the tested buildings, and the detection rate of cracks reached 91.67%. Figure 13 shows the visualization of the crack width measurement results, where the actual width of all cracks and the measured width in both detection cases are shown in Figure 13a, and Figure 13b shows the absolute error of all cracks in both detection cases.

This experiment shows that the crack information-assisted drone flight automatic control algorithm proposed in this paper can effectively drive the drone close to the cracks on the building surface and obtain significantly better results of crack geometry edge extraction as well as crack width measurement.

However, a few millimeter-scale cracks still have not been successfully detected in this experiment, mainly due to the difference between the actual engineering environment and the model training dataset. In addition, through the visualization of the detection results in Figure 13, it can be seen that the measured width of most cracks is larger than the actual width, partly due to the error in the depth sensing of the binocular camera. Another primary reason is that the high-precision crack segmentation algorithm tends to predict the width of cracks much wider when shooting farther away.

3.3 | Overall method experiment on the shaking table test model

When performing crack detection on the surface of an actual building, the drone cannot be too close to the building surface to prevent a collision. In addition, the drone

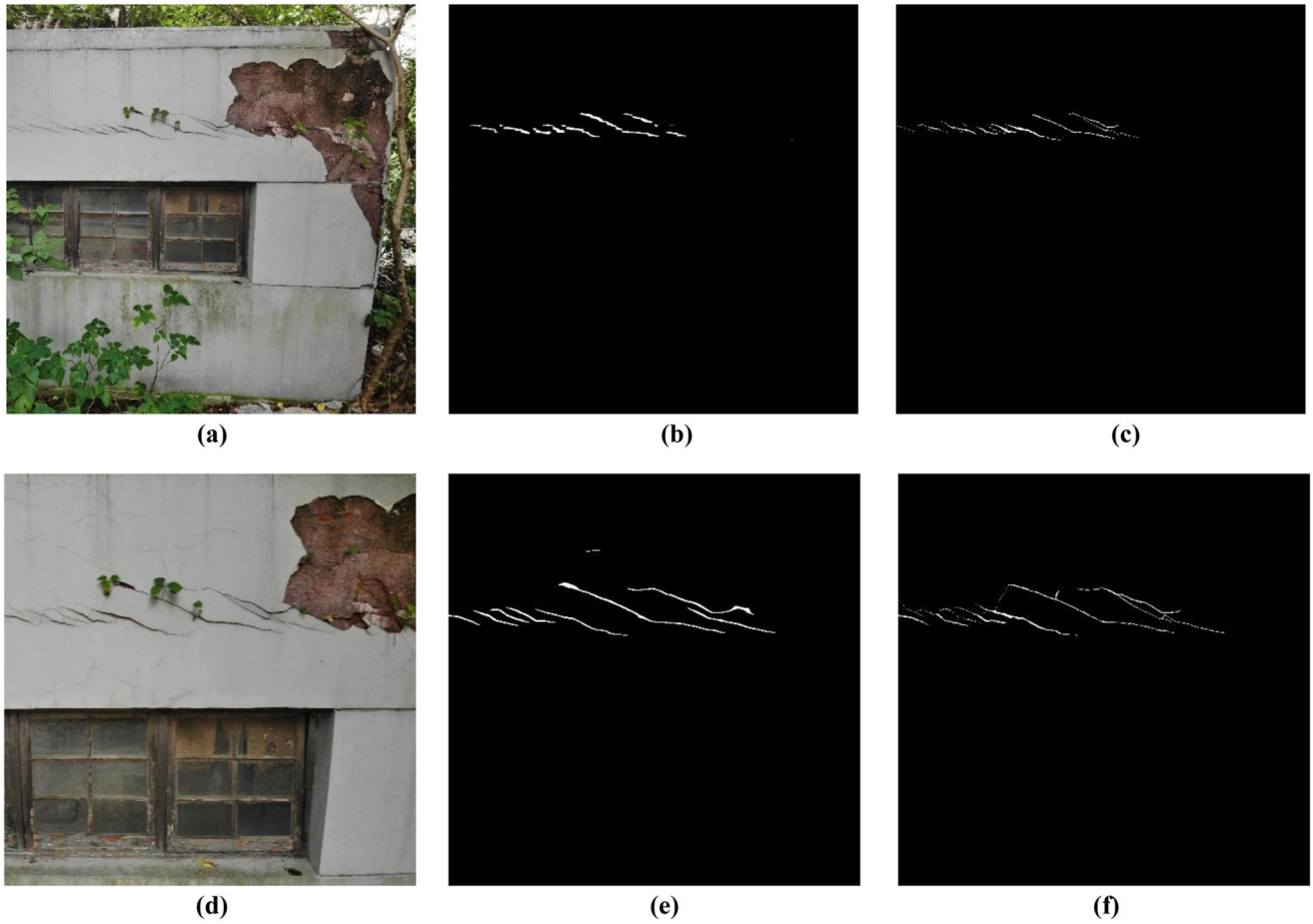

FIGURE 12 Visualization of crack edge extraction results in two cases: (a), (b), and (c) show the original image, detection result, and ground truth when the drone is not close to the building surface, respectively, (d), (e) and (f) show the original image, detection result, and ground truth when the drone approaches the building surface through the crack information-assisted drone flight automatic control algorithm, respectively.

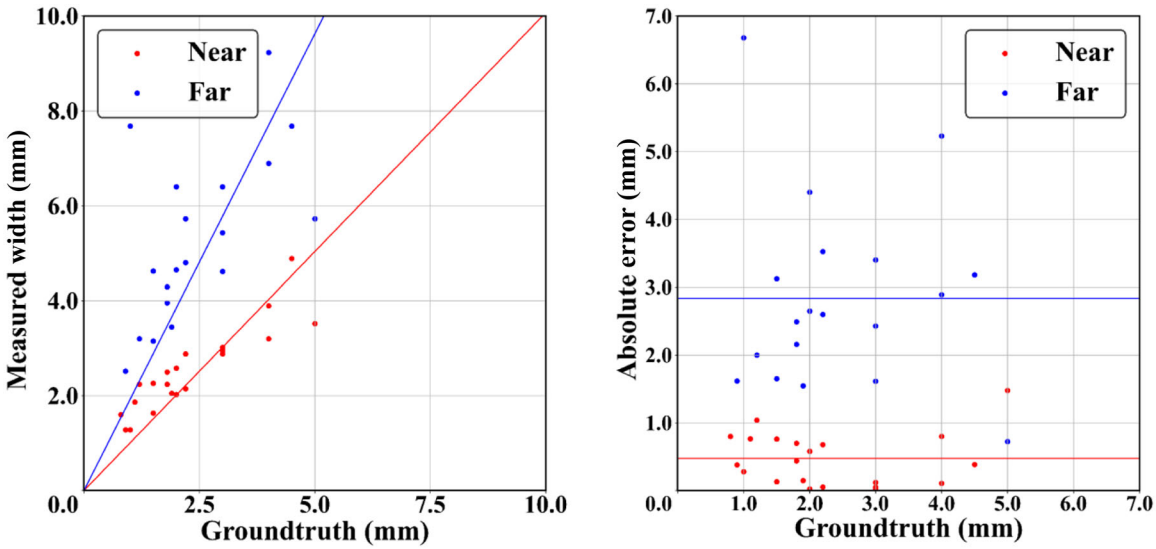

FIGURE 13 Visualization of the crack width measurement results: (a) shows the actual width of all cracks and measured width with and without the crack information-assisted drone flight automatic control algorithm, (b) shows the absolute error of crack width measurement with and without the algorithm. Among them, the "near" in the legend represents the detection result using the algorithm, and the "far" means the detection result without using the algorithm.

has turbulence during the flight, and the distance between the camera and the crack is difficult to calibrate in actual working conditions. Therefore, in practical engineering, the detection accuracy of crack width based on the drone is difficult to achieve the detection accuracy under ideal shooting conditions.

This experiment aims to verify that in the actual working conditions, the method proposed in this paper can achieve high-precision crack width measurement in the presence of the above difficulties. Therefore, a shaking table test model was selected as the test building. In this experiment, the flight trajectory of the drone is roughly planned through fixed-point navigation rather than detailed planning. This is consistent with the situation that it is impossible to carry out refined route planning for each building when a large-scale inspection of buildings is performed in actual working conditions. Therefore, the crack width measurement results in this experiment demonstrate the detection accuracy of the proposed method in complex environments. Furthermore, the results of the maximum crack width measurement are compared for two cases, namely, with and without using the crack information-assisted drone flight automatic control algorithm. Thus, the effectiveness of the method proposed in this paper is verified.

Besides, the actual maximum crack width of this experiment is obtained by manual measurement, and the tool used is a crack measuring ruler. Since the drone cannot enter the interior of the building, all measurements were made of surface cracks. This paper aims to perform millimeter-scale crack detection, which is also the crack width that needs to be focused on in actual applications. Only cracks with a width of about 1 mm or more need to be detected. Therefore, other cracks with a maximum width of less than 0.5 mm are not considered in the detection range.

3.3.1 | Introduction of the building

The building used in this experiment is a shaking table test model of synchronous implementation of an engineering project with a prefabricated slab of rail transit line 11 in Chisha, Guangzhou. The original structure has a total height of 161.5 m, including a three-story podium with a total height of 29.2 m (including an underground floor). The upper residential tower adopts a shear wall structure. The overall structure system is a partially frame-supported shear wall structure (the transfer floor is a three-story podium). The plan size of the residential tower is 38.9 × 32.8 m, and the plan size of the podium is about 68.95 × 81.60 m. The shaking table test model adopts a scale of 1:10. The scaled model is shown in Figure 14.

FIGURE 14 Shaking table test model

3.3.2 | Experiment and result

The method proposed in this paper is used to conduct automated crack detection on this building, in which the flight of the drone is realized through fixed-point navigation. In the crack detection on the surface of one side of the building, the drone's route is planned through fixed-point navigation to realize the automatic crack detection combined with the crack information-assisted drone flight automatic control algorithm. Since there are no visible millimeter-scale cracks in the surfaces of the remaining three directions, these areas are not covered by the drone's fixed-point navigation. In the fixed-point navigation, 37 control points are specified, and the distance between the control point and the building is 2 m. The reason for choosing this distance is to simulate the actual detection situation. If the drone is too close to the building for a long time, it is extremely prone to collision. The schematic diagram of the specific location of the control points and the flight trajectory of the drone is shown in Figure 15. The position indicated by the red arrow in Figure 15 is the position where the drone automatically detected the crack. In this experiment, the parameters of the crack information-assisted drone flight automatic control algorithm are the same as

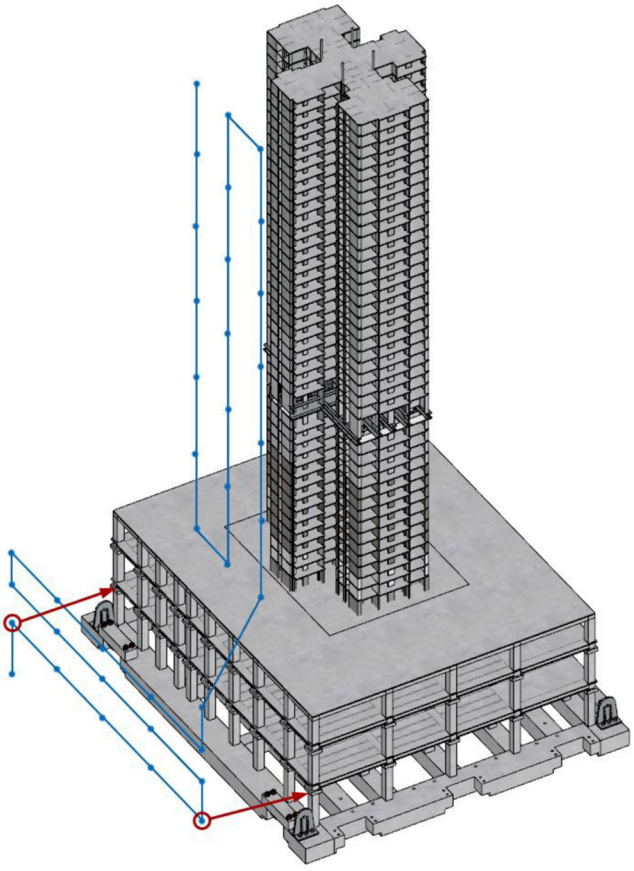

FIGURE 15 The visualization of the trajectory of the fixed-point navigation and the location of the crack detected by the drone

those in Section 3.2, except that the target shortest distance d_t is set to 0.8 m. Furthermore, as in Section 3.2.2, the crack detection model used in this experiment is trained using the SBGCrack dataset and does not exploit prior information in the experimental setting. Higher detection accuracy can be achieved if the model is trained using crack data collected in the experimental environment.

Since the building in this experiment is a scaled shaking table test model, and due to the relatively large size of the drone equipment, the drone cannot enter the interior of the building. Only visible cracks on the building surface are therefore automatically detected. The experimental results show that the buildings used in this experiment were less damaged and had no visible cracks in the upper half of the structure. Additionally, several visible cracks are successfully detected on the columns in the lower half of the building. On the two columns at the bottom of the building, the drone successfully detected millimeter-level cracks and automatically approached about 1 m from the surface of the building by utilizing the crack information-assisted drone flight automatic control algorithm, thus realizing more accurate crack width measurement. In this experiment, all cracks on the surface of the building with a maximum width greater than 0.5 mm were successfully detected.

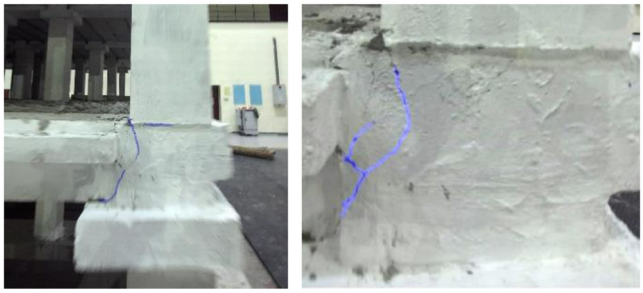

FIGURE 16 The visualization of the comparison of the crack edge detection results in two cases: (a) shows the detection results when approaching the building surface using the crack information-assisted drone flight automatic control algorithm, and (b) shows the detection results when the algorithm is not utilized

In order to verify the effectiveness of the crack information-assisted drone flight automatic control algorithm, the crack width was extracted from the images taken by the drone before approaching the building surface, and it was compared with the final result of the crack width measurement. It should be noted that the algorithms used for crack width measurement in the above two cases are the same high precision segmentation model and maximum crack width measurement algorithm introduced in Section 2. The detection results of the maximum crack width in the two cases are shown in Table 6, and the visualization of the comparison of the crack edge detection result is shown in Figure 16. In this experiment, the travel distance of the drone is 78.705 m, and the total scanning time is 186 s. All crack detection calculations are completed during the movement of the drone. In this experiment, the recall, precision, and MIoU of crack edge extraction after the drone is close to the building are 63.26%, 59.11%, and 44.00%, respectively, and only 48.88%, 26.31%, and 20.63% when the crack information-assisted drone flight automatic control algorithm is not used.

The experimental results show that the average absolute error of detecting the maximum crack width through the drone at 2 m from the building surface is 2.89 mm. When the drone is close to the building surface by implementing the crack information-assisted drone flight automatic control algorithm, the average absolute error of maximum crack width measurement is only 0.42 mm. This is because when the drone is far away from the building, the calculation error of the pixel resolution is larger, and the high precision segmentation model has a lower detection accuracy for the extraction of the edge of the crack. However, in this experiment, since the drone cannot safely navigate inside the shaking table test building, it can only

TABLE 6 The predicted and the actual crack width of the shaking table test model (mm)

Crack number	Real crack width	Predicted crack width	Error	Use drone flight automatic control algorithm
1	1.0	1.53	0.53	Yes
2	0.5	0.98	0.48	Yes
3	0.8	1.05	0.25	Yes
1	1.0	3.95	2.95	No
2	0.5	3.42	2.92	No
3	0.8	3.59	2.79	No

effectively detect the cracks on the outer surface of the building instead of all the cracks in the building.

In summary, the effectiveness of the crack information-assisted drone flight automatic control algorithm and the automatic maximum crack width measurement accuracy is verified. In addition, the effectiveness of the method proposed in this paper has also been verified in practical situations.

4 | CONCLUSION

This paper proposes an automatic crack detection method based on a drone, including a lightweight crack segmentation algorithm, a lightweight crack segmentation algorithm, a high-precision crack segmentation algorithm, a maximum crack width measurement algorithm, and a crack information-assisted drone flight automatic control algorithm. Using the method proposed in this paper can assist the drone in path planning to approach the crack during the crack detection process and achieve safer, more efficient, and more accurate crack detection. Based on the present investigation, the conclusion is summarized as follows:

1. Through the crack detection method proposed in this paper, crack detection can be completed in real time during the drone's flight, thereby improving the efficiency of crack detection. By deploying the proposed lightweight crack classification algorithm and lightweight crack segmentation algorithm on the drone, crack detection with an inference speed of tens of milliseconds can be easily performed. Thus, the critical decision-making information required for the path planning of the drone can be provided in time. In addition, the high-precision crack segmentation algorithm proposed in this paper can further improve the MIoU of crack edge detection without affecting the subsequent real-time crack detection.
2. The proposed crack information-assisted drone flight automatic control algorithm combines the crack detection results in the images with the flight control system of the drone, allowing the drone to approach the crack area automatically and improve the crack detection results with higher accuracy. The proposed method is tested on a two-story building and a large shaking table test building. Observation of the flight path and captured images proved that it is possible to automatically locate the crack region and control the drone near the crack. In addition, the detection results of two experiments show that by using the crack information-assisted drone flight automatic control algorithm proposed in this paper, the MIoU of crack geometric edge detection and the accuracy of crack maximum width measurement are greatly improved. The effectiveness of the crack information-assisted drone flight automatic control algorithm has been verified;
3. A maximum crack width measurement algorithm has been proposed to obtain a quantitative measure of crack width. The proposed method is verified on a large shaking table test building. In the non-ideal shooting environment based on a drone and the actual shooting environment without distance calibration, the absolute error of crack width measurement results on the two-story building and the shaking table test building is 0.48 mm and 0.4 mm, respectively, which shows that the method proposed in this paper has the ability to detect small cracks in practical and complex situations.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial support from the Distinguished Young Scientist Fund of National Natural Science Foundation of China (Grant No. 52025083).

REFERENCES

Alaknanda, Anand, R. S., & Kumar, P. (2009). Flaw detection in radiographic weldment images using morphological watershed segmentation technique. *Ndt & e International*, *42*(1), 2–8.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12), 2481–2495.

Bao, Y., Li, J., Nagayama, T., Xu, Y., Spencer, B. F., & Li, H. (2021). The 1st international project competition for structural health monitoring (IPC-SHM, 2020): A summary and benchmark problem. *Structural Health Monitoring*, *20*(4), 2229–2239.

Bouguet, J.-Y., & Perona, P. (1998). Camera calibration from points and lines in dual-space geometry. *Proceedings of the 5th European Conference on Computer Vision*, Freiburg, Germany (pp. 2–6).

Çelik, F., & König, M. (2022). A sigmoid-optimized encoder–decoder network for crack segmentation with copy-edit-paste transfer learning. *Computer-Aided Civil and Infrastructure Engineering*, *37*(14), 1875–1890.

Cha, Y.-J., Choi, W., Suh, G., Mahmoudkhani, S., & Büyüköztürk, O. (2018). Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. *Computer-Aided Civil and Infrastructure Engineering*, *33*(9), 731–747.

Chen, F.-C., & Jahanshahi, M. R. (2018). NB-CNN: Deep learning-based crack detection using convolutional neural network and naïve Bayes data fusion. *IEEE Transactions on Industrial Electronics*, *65*(5), 4392–4400.

Chen, J., & He, Y. (2022). A novel U-shaped encoder–decoder network with attention mechanism for detection and evaluation of road cracks at pixel level. *Computer-Aided Civil and Infrastructure Engineering*, *37*(13), 1721–1736.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018a). Encoder-decoder with atrous separable convolution for semantic image segmentation. *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany (pp. 801–818).

Choi, S.-S., & Kim, E.-K. (2015). Building crack inspection using small uav. *2015 17th International Conference on Advanced Communication Technology (ICACT)*, Pyeongchang, Korea (pp. 235–238).

Deng, J., Lu, Y., & Lee, V. C.-S. (2020). Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network. *Computer- Aided Civil and Infrastructure Engineering*, *35*(4), 373–388.

Dung, C. V., & Anh, L. D. (2019). Autonomous concrete crack detection using deep fully convolutional neural network. *Automation in Construction*, *99*, 52–58.

Farbman, Z., Fattal, R., Lischinski, D., & Szeliski, R. (2008). Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics (TOG)*, *27*(3), 1–10.

Fei, Y., Wang, K. C., Zhang, A., Chen, C., Li, J. Q., Liu, Y., Yang, G., & Li, B. (2019). Pixel-level cracking detection on 3d asphalt pavement images through deep-learning-based cracknet-v. *IEEE Transactions on Intelligent Transportation Systems*, *21*(1), 273–284.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition*, Las Vegas, NV (pp. 770–778).

Hirschmuller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(2), 328–341.

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. *Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, Salt* Lake City, UT (pp. 7132–7141).

Iyer, S., & Sinha, S. K. (2006). Segmentation of pipe images for crack detection in buried sewers. *Computer-Aided Civil and Infrastructure Engineering*, *21*(6), 395–410.

Jiang, S., & Zhang, J. (2020). Real-time crack assessment using deep neural networks with wall-climbing unmanned aerial system. *Computer-Aided Civil and Infrastructure Engineering*, *35*(6), 549–564.

Jin, S., Lee, S. E., & Hong, J.-W. (2020). A vision-based approach for autonomous crack width measurement with flexible kernel. *Automation in Construction*, *110*, 103019.

Kamaliardakani, M., Sun, L., & Ardakani, M. K. (2016). Sealed-crack detection algorithm using heuristic thresholding approach. *Journal of Computing in Civil Engineering*, *30*(1), 04014110.

Kang, D., & Cha, Y.-J. (2018). Autonomous uavs for structural health monitoring using deep learning and an ultrasonic beacon system with geo-tagging. *Computer-Aided Civil and Infrastructure Engineering*, *33*(10), 885–902.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kong, S.-Y., Fan, J.-S., Liu, Y.-F., Wei, X.-C., & Ma, X.-W. (2021). Automated crack assessment and quantitative growth monitoring. *Computer-Aided Civil and Infrastructure Engineering*, *36*(5), 656–674.

Kong, X., & Li, J. (2018). Vision-based fatigue crack detection of steel structures using video feature tracking. *Computer-Aided Civil and Infrastructure Engineering*, *33*(9), 783–799.

Lai, M. (2015). Deep learning for medical image segmentation. arXiv:1505.02000 [cs].

Li, S., Zhao, X., & Zhou, G. (2019). Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network. *Computer-Aided Civil and Infrastructure Engineering*, *34*(7), 616–634.

Liang, X. (2019). Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization. *Computer-Aided Civil and Infrastructure Engineering*, *34*(5), 415–430.

Liu, C., & Xu, B. (2022). A night pavement crack detection method based on image-to-image translation. *Computer-Aided Civil and Infrastructure Engineering*, *37*(13), 1737–1753.

Liu, J., Yang, X., Lau, S., Wang, X., Luo, S., Lee, V. C.-S., & Ding, L. (2020). Automated pavement crack detection and segmentation based on two-step convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering*, *35*(11), 1291–1305.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, 3431–3440.

Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. *Proceedings of the European Conference On Computer Vision (ECCV)*, Munich, Germany (pp. 116–131).

Meng, S., Gao, Z., Zhou, Y., He, B., & Kong, Q. (2022). A three-stage deep-learning-based method for crack detection of high resolution steel box girder image. *Smart Structures and Systems*, *29*(1), 29–39.

Meng, S., Zhang, X., Qiao, S., & Zhou, Y. (2020). Research on grid optimized crack detection model based on deep learning. *Journal of Building Structures*, *41*(S2), 404–410.

Mohan, A., & Poobal, S. (2018). Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*, *57*(2), 787–798.

Nayyeri, F., Hou, L., Zhou, J., & Guan, H. (2019). Foreground–background separation technique for crack detection. *Computer-Aided Civil and Infrastructure Engineering*, *34*(6), 457–470.

Ni, F., Zhang, J., & Chen, Z. (2019). Zernike-moment measurement of thin-crack width in images enabled by dual-scale deep learning. *Computer-Aided Civil and Infrastructure Engineering*, *34*(5), 367–384.

Özgenel, F. (2019). *Concrete crack segmentation dataset*. Mendeley Data. https://data.mendeley.com/datasets/jwsn7tfbrp/1

Pan, X., & Yang, T. (2020). Postdisaster image-based damage detection and repair cost estimation of reinforced concrete buildings using dual convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering*, *35*(5), 495–510.

Pan, Y., Zhang, G., & Zhang, L. (2020). A spatial-channel hierarchical deep learning network for pixel-level automated crack detection. *Automation in Construction*, *119*, 103357.

Qu, Z., Lin, L.-D., Guo, Y., & Wang, N. (2015). An improved algorithm for image crack detection based on percolation model. *IEEJ Transactions on Electrical and Electronic Engineering*, *10*(2), 214–221.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. *Paper presented at International Conference on Medical Image Computing and Computer-Assisted Intervention*. Cham, Switzerland: Springer, 234–241.

Salih, A. L., Moghavvemi, M., Mohamed, H. A., & Gaeid, K. S. (2010). Flight PID controller design for a UAV quadrotor. *Scientific Research and Essays*, 5(23), 3660–3667.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition*, Salt Lake City, UT (pp. 4510–4520).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, May 7–9.

Sirca, G. F. Jr., & Adeli, H. (2018). Infrared thermography for detecting defects in concrete structures. *Journal of Civil Engineering and Management*, *24*(7), 508–515.

Song, W., Jia, G., Jia, D., & Zhu, H. (2019). Automatic pavement crack detection and classification using multiscale feature attention network. *IEEE Access*, *7*, 171001–171012.

Sun, B.-C., & Qiu, Y.-J. (2007). Automatic identification of pavement cracks using mathematic morphology. In Q. Peng, K. Wang, Y. Qui, X. Luo, & B. Shuai (Eds.), *International Conference on Transportation Engineering 2007* (pp. 1783–1788). ASCE.

Wang, W., Zhang, A., Wang, K. C., Braham, A. F., & Qiu, S. (2018). Pavement crack width measurement based on Laplace's equation for continuity and unambiguity. *Computer-Aided Civil and Infrastructure Engineering*, *33*(2), 110–123.

Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). Cbam: Convolutional block attention module. *Proceedings of the European Conference on Computer Vision (ECCV)*, Salt Lake City, UT (pp. 3–19).

Yang, X., Li, H., Yu, Y., Luo, X., Huang, T., & Yang, X. (2018). Automatic pixel-level crack detection and measurement using fully convolutional network. *Computer-Aided Civil and Infrastructure Engineering*, *33*(12), 1090–1109.

Ye, X.-W., Jin, T., & Chen, P.-Y. (2019). Structural crack detection using deep learningbased fully convolutional networks. *Advances in Structural Engineering*, *22*(16), 3412–3419.

Yoon, H., Shin, J., & Spencer, B. F. Jr. (2018). Structural displacement measurement using an unmanned aerial system. *Computer-Aided Civil and Infrastructure Engineering*, *33*(3), 183–192.

Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., & Sang, N. (2020). BiSeNet V2: Bilateral network with guided aggregation for real-time semantic segmentation. *International Journal of Computer Vision, 2021*, *129*(11), 3051–3068.

Zhang, A., Wang, K. C., Li, B., Yang, E., Dai, X., Peng, Y., Fei, Y., Liu, Y., Li, J. Q., & Chen, C. (2017). Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network. *Computer-Aided Civil and Infrastructure Engineering*, *32*(10), 805–819.

Zhang, A., Wang, K. C. P., Fei, Y., Liu, Y., Tao, S., Chen, C., Li, J. Q., & Li, B. (2018). Deep learning-based fully automated pavement crack detection on 3D asphalt surfaces with an improved CrackNet. *Journal of Computing in Civil Engineering*, *32*(5), 0401 8041.

Zhang, T. Y., & Suen, C. Y. (1984). A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, *27*(3), 236–239.

Zhang, Y., & Yuen, K.-V. (2021). Crack detection using fusion features-based broad learning system and image processing. *Computer-Aided Civil and Infrastructure Engineering*, *36*(12), 1568–1584.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(11), 1330–1334.

Zhao, H., Qin, G., & Wang, X. (2010). Improvement of canny algorithm based on pavement edge detection. *2010 3rd International Congress On Image And Signal Processing*, Yantai, China (pp. 964–967).

Zhou, L., Zhang, C., & Wu, M. (2018). D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Salt Lake City, UT (pp. 182–186).

Zhou, Y., & Liu, T. (2019). Computer vision-based crack detection and measurement on concrete structure. *Journal of Tongji University (Natural Science)*, *47*(9), 1277–1285.

Zou, D., Zhang, M., Bai, Z., Liu, T., Zhou, A., Wang, X., Cui, W., & Zhang, S. (2022). Multicategory damage detection and safety assessment of post-earthquake reinforced concrete structures using deep learning. *Computer-Aided Civil and Infrastructure Engineering*, *37*(9), 1188–1204.

How to cite this article: Meng, S., Gao, Z., Zhou, Y., He, B., & Djerrad, A. (2023). Real-time automatic crack detection method based on drone. *Computer-Aided Civil and Infrastructure Engineering*, *38,* 849–872. https://doi.org/10.1111/mice.12918