CONCRETE CRACK DETECTION AND QUANTIFICATION USING COMPUTER VISION

Project-1 (CE47005) Report
submitted to
Indian Institute of Technology Kharagpur
in partial fulfillment for the award of the degree of

B.Tech - M.Tech (Dual Degree 5Y)
in

CIVIL ENGINEERING/ARTIFICIAL INTELLIGENCE
MACHINE LEARNING AND APPLICATIONS

By
KOLATHURU VISHNU VARDHAN
(21CE3AI13)

Under the supervision of
PROF. DAMODAR MAITY

DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR
AUTUMN SEMESTER, 2024-25

NOVEMBER 28, 2024

DECLARATION

I certify that

1. The work contained in this report has been done by me under the guidance of my supervisor.
2. The work has not been submitted to any other Institute for any degree or diploma.
3. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
4. Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Kolathuru Vishnu Vardhan
21CE3AI13

Place: Kharagpur
Date: 28th November, 2024

DEPARTMENT OF CIVIL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled **Concrete Crack Detection and Quantification Using Computer Vision** submitted by **Kolathuru Vishnu Vardhan** (Roll No. 21CE3AI13) an undergraduate student of **Department of CIVIL ENGINEERING/ARTIFICIAL INTELLIGENCE MACHINE LEARNING AND APPLICATIONS** towards partial fulfillment for the award of degree of B.Tech-M.Tech (Dual Degree 5Y) in CIVIL ENGINEERING/ARTIFICIAL INTELLIGENCE MACHINE LEARNING AND APPLICATIONS is a record of bonafide work carried out by him under my supervision and guidance during Autumn semester, 2024-25.

Prof. Damodar Maity
Department of Civil Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Place: Kharagpur
Date: 28th November, 2024

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Prof. Damodar Maity, my project supervisor, for his invaluable guidance and support throughout the course of this project. His expertise and direction played a crucial role in navigating challenges and achieving the project objectives.

I am also thankful to Mr. Saurabh Kumar Singh, a Ph.D. scholar in the Department of Civil Engineering, for his mentorship. His insightful discussions and valuable inputs greatly contributed to the technical depth and overall quality of this work.

Finally, I would like to acknowledge the support of my family and friends, whose encouragement has been instrumental in the successful completion of this project.

ABSTRACT

Detecting and quantifying cracks in structures are essential for assessing their structural health and ensuring safety. In this study, we present a comprehensive approach for crack detection and quantification using the YOLOv11 deep learning framework. The proposed method is designed to identify, localize, and quantify cracks in concrete structures. Crack detection is performed by training YOLOv11 on a dataset of 1,887 annotated images, augmented to a total of 3,287 images to enhance robustness. The dataset is split into 85% for training (2,800 images) and 15% for validation (487 images). Images were preprocessed using CLAHE, Otsu Thresholding, and noise reduction techniques to improve crack visibility. Training was conducted with an input size of 256 × 256 pixels, a batch size of 64, and extended fine-tuning for 150 epochs.

The YOLOv11 model achieved a mean average precision (mAP_0.5) of 92%, mAP_0.5:0.95 of 77%, and an F1 score of 83%, demonstrating its effectiveness in detecting cracks. Beyond detection, the model quantifies crack dimensions, including length, width, perimeter, and area, focusing on high-confidence detections. This automated pipeline can significantly improve real-time structural health monitoring by accurately detecting and quantifying cracks, enabling timely maintenance of critical infrastructure.

Table of Contents

Chapter 1

1.1 Introduction

Crack formation and propagation in concrete significantly compromise the structural integrity and durability of civil infrastructure. Cracks provide pathways for aggressive agents, such as water and chlorides, to corrode steel reinforcements, reducing the load-bearing capacity of concrete members. Over time, this leads to structural deterioration and potential failure, highlighting the critical need for timely crack identification during maintenance and inspection.

Causes of Crack Formation in Concrete Structures

1. Excessive water in the concrete mix weakens the material, making it prone to cracking during shrinkage.
2. Shrinkage stresses occur when concrete shrinks as it cures, causing cracks if the strain exceeds the material's strength.
3. Soil movement and expansion can exert pressure, leading to vertical cracks in foundations and walls.
4. Seasonal thermal effects, like cyclical temperature changes, cause expansion and contraction, creating internal stresses and cracks.
5. Corrosion of reinforcement caused by marine salts or chemicals increases the volume of reinforcement, creating pressure that results in cracks.

Traditional crack detection methods, such as manual inspection, are subjective and dependent on the inspector's expertise. While image processing techniques (IPTs) like thresholding and edge detection have been explored, they remain limited by environmental factors like lighting and surface roughness. Machine learning (ML) and deep learning (DL) methods have gained prominence for crack detection. ML-based approaches, such as support vector machines (SVMs) and artificial neural networks (ANNs), rely on handcrafted features for crack identification. However, these methods often struggle with false positives due to irrelevant attributes extracted from the images. Moreover, many ML approaches focus solely on detecting the presence of cracks without assessing their severity or dimensions, limiting their utility for structural health monitoring.

Deep learning models, particularly convolutional neural networks (CNNs) and YOLO (You Only Look Once), have shown superior performance in automating feature extraction and minimizing false positives. YOLO-based models are particularly effective for detecting and localizing cracks in real-time. Despite advances, many crack detection systems still face challenges with dataset size, computational requirements, and robustness in varying environmental conditions.

This study proposes a two-stage pipeline:

1. **Crack Detection and Localization:** Using YOLOv11 to accurately detect and localize cracks under diverse conditions.
2. **Crack Quantification:** Applying enhanced image preprocessing techniques (CLAHE, Otsu Thresholding, noise reduction) to measure crack dimensions (length, width, perimeter, area) for high-confidence detections.

1.2 Objective

Crack detection in concrete is crucial for maintaining safety and structural integrity. While visual inspections can spot cracks, they are time-consuming, subjective, and constrained by limited human resources. Automated systems using machine learning and deep learning, like YOLOv11, provide a faster, more accurate alternative. The following points outline the key objectives of this project and its importance:

1. Manual crack detection is time-consuming, subjective, and error-prone. This project automates the process using YOLOv11, improving accuracy and efficiency.
2. Large-scale infrastructure requires extensive manpower for inspection, which is limited by time. The system can analyze large datasets quickly, offering a scalable solution.
3. The algorithm not only detects cracks but also quantifies their severity by measuring dimensions, helping prioritize repairs based on the level of damage.
4. Manual methods often delay identifying critical cracks. Automated detection provides real-time results, enabling faster responses and proactive maintenance.
5. Human assessment is inconsistent and labor-intensive. Automated detection reduces human dependency, ensuring reliable, repeatable evaluations and optimal resource use.

1.3 Literature Review

1. **Automated pixel-level crack detection and quantification using deep convolutional neural networks for structural condition assessment**
   The R-FPANet model utilizes ResNet-50 for feature extraction and a Feature Pyramid Network with Dense Blocks (FPN-DB) for multi-scale feature extraction, enhanced by self-attention mechanisms. The model demonstrated remarkable performance, achieving an Intersection over Union (IoU) score of 83.07%. This approach excels at detecting cracks and quantifying parameters such as area, length, and width.
   Limitation: While effective, the model may struggle with real-time applications and large datasets in complex environments, a challenge that YOLOv11 addresses with its efficient processing and scalability.

2. **Multi-task deep learning model for detecting and quantifying cracks in reinforced concrete (RC) structures**
   This study proposes a multi-task model combining crack segmentation and centerline prediction, improving crack width measurement and using 3D reconstruction to correct image distortions.
   Limitation: The model performs well under controlled conditions but struggles with generalization in varied environmental factors like lighting and background interference. YOLOv11 offers better robustness across diverse conditions, ensuring more reliable real-time monitoring.

3. **Computer vision-based crack detection and quantification methodologies for civil structures**
   This review systematically compares supervised and unsupervised learning methods, highlighting challenges such as limited training datasets, background interference, and generalization issues across different structural materials.
   Limitation: Many models fail to address these challenges effectively. YOLOv11 overcomes these limitations by using more diverse and robust datasets, ensuring better adaptability to a range of structural conditions.

4. **Two-stage Method Based on the You Only Look Once Framework and Image Segmentation for Crack Detection in Concrete Structures**
   This approach uses a two-stage methodology: the first stage detects and localizes cracks using YOLOv5, while the second stage focuses on precise dimension measurement.
   Limitation: YOLOv5 performs well in crack detection but lacks sufficient precision in crack quantification. The YOLOv11 model improves upon this by incorporating advanced preprocessing techniques like CLAHE, Otsu Thresholding, and noise reduction for more accurate crack measurements.

Chapter 2

2.1 Proposed model for the detection of cracks and determination of crack properties

YOLOv11 is an advanced object detection model optimized for real-time crack detection and quantification, improving upon earlier YOLO versions. It combines speed and accuracy through innovative architectural components designed to handle diverse and challenging crack patterns.

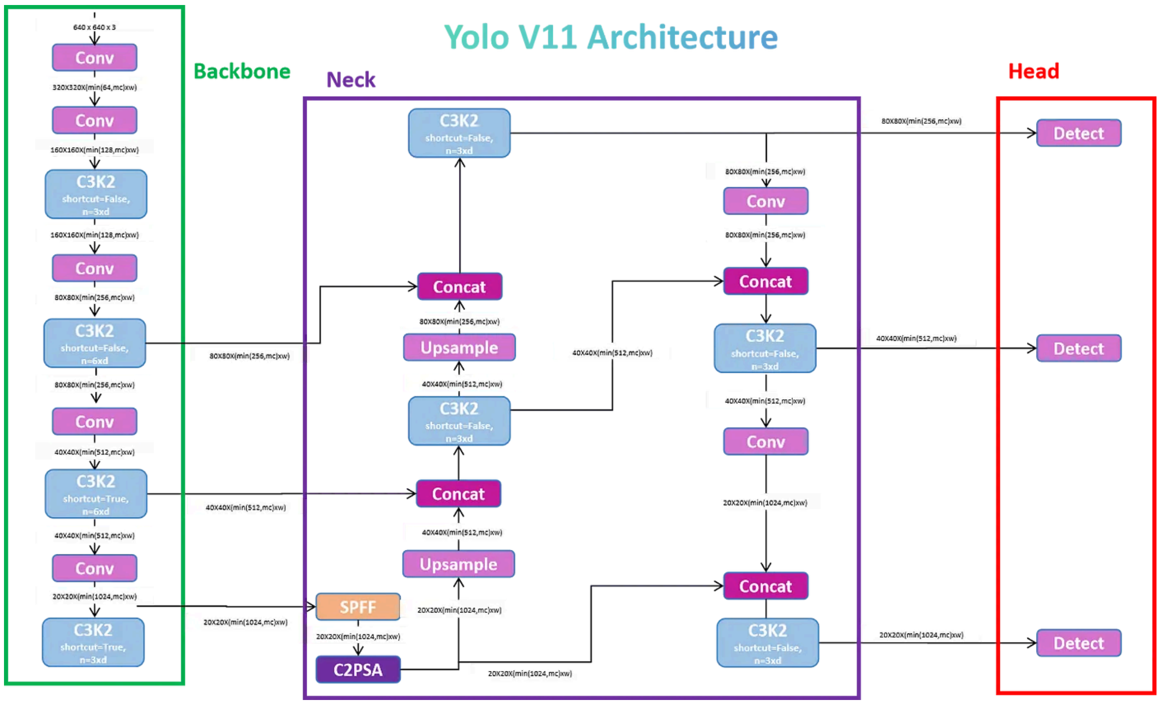


Fig. 1 Architecture of the YOLO11

1. **Backbone:**
    - **Convolutional Block:** Processes input features using 2D convolution, batch normalization, and SiLU activation, enhancing non-linearity learning and gradient flow.
    - **Bottle Neck Layer:** Residual connections inspired by ResNet preserve crucial information in deep networks, improving feature extraction efficiency.
    - **C2F Block:** The Cross Stage Partial Focus block divides feature maps into two streams processed through bottleneck layers, minimizing redundancy while enhancing feature preservation.
    - **C3K2 Block:** An enhancement over the C2F block, it employs smaller 3x3 convolutional kernels to balance speed and accuracy, optimizing information flow with fewer parameters.

2. **Neck:**
    - **Spatial Pyramid Pooling Fast (SPFF):** Aggregates multi-scale contextual features via max-pooling at different scales, allowing effective detection of cracks of varying sizes. This ensures robust performance on both large and small-scale cracks.
3. **Attention Mechanism:**
    - **C2PSA Block:** Incorporates position-sensitive attention to focus on spatially significant features, improving detection of small, thin, or partially occluded cracks.
    - **Position-Sensitive Attention (PSA):** Further enhances spatial attention by highlighting critical regions in feature maps, enabling the detection of fine details in complex environments.
4. **Head:**
   A multi-scale detection head operates on three levels (P3, P4, P5) to detect cracks of varying sizes and scales. By processing features at low, medium, and high granularity, it ensures precise localization of cracks, outputting bounding boxes and class labels for all detected objects.

2.2 Image Preprocessing and Segmentation

Effective preprocessing and segmentation are critical for accurate crack detection and quantification. In this project, techniques like contrast enhancement, adaptive thresholding, and image cleaning were implemented to improve the quality and reliability of input data for the YOLOv11 model.

1. **Contrast Enhancement**
   Contrast Limited Adaptive Histogram Equalization (CLAHE) was used to improve the visibility of cracks in grayscale images. By dividing the image into small tiles and applying histogram equalization to each, CLAHE enhances local contrast and mitigates issues such as uneven illumination and low brightness. This technique ensures that fine crack details are more distinguishable, even under challenging lighting conditions.
2. **Otsu Thresholding**
   Otsu's adaptive thresholding was employed to binarize the images by separating crack regions from the background. The algorithm automatically determines the optimal threshold by minimizing intra-class variance in pixel intensity. For images with complex backgrounds or uneven lighting, manual threshold adjustments were applied to refine the segmentation. After thresholding, morphological operations like opening were used to remove noise and produce cleaner binary images.
3. **Image Cleaning**
   Post-segmentation, irrelevant or noisy objects were filtered out to retain only meaningful crack features. This was achieved using size filtering and the Axis Ratio Index (ARI).

Small objects below a specified pixel threshold were discarded, as they were unlikely to represent cracks. ARI, defined as the ratio of the major axis length to the minor axis length of an object, was used to identify elongated crack-like structures. Objects with ARI values below a certain threshold were removed, ensuring only relevant structures were preserved.

Significance

These preprocessing and segmentation steps enhanced the quality of input data for YOLOv11. By improving contrast, accurately segmenting crack regions, and removing noise, the model was able to detect and quantify cracks with high precision, addressing the challenges posed by real-world scenarios effectively.

2.3 Workflow

The flowchart of the proposed method is illustrated in Fig.2. The proposed method involves data collection, data preprocessing, data annotation, data augmentation, model training and evaluation.

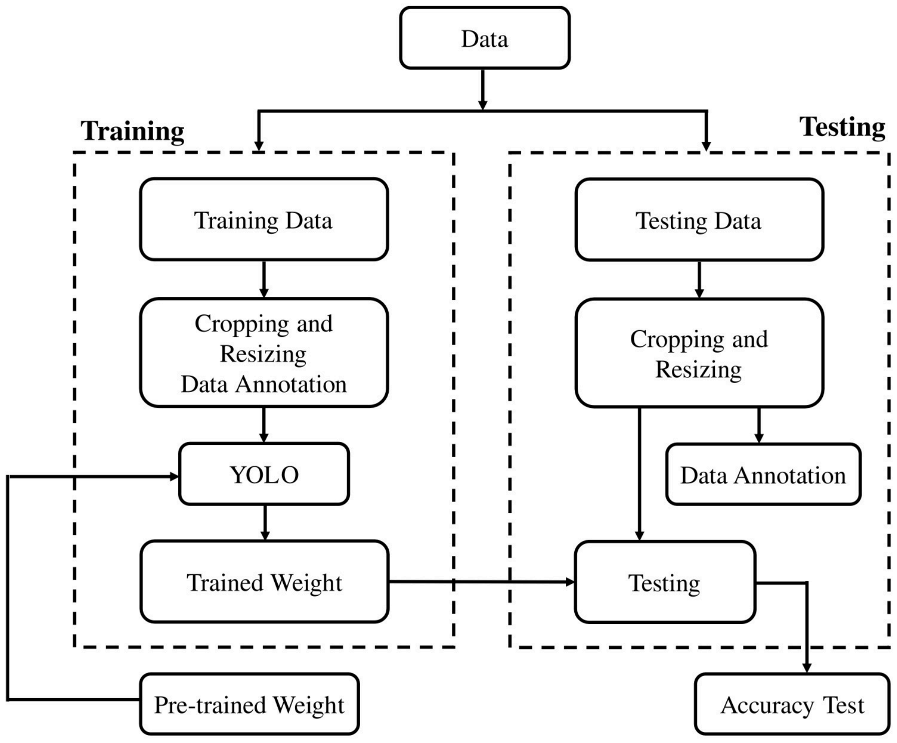


Fig. 2 Flowchart of the methodology used in this study

1. Data Collection

The dataset for crack detection was sourced from the public METU image database, which contains two classes: cracked and uncracked areas. The database includes a total of 40,000 images (227 × 227 pixels), with 20,000 images per class. For training the YOLOv11 model, a subset of 1,887 images was selected.

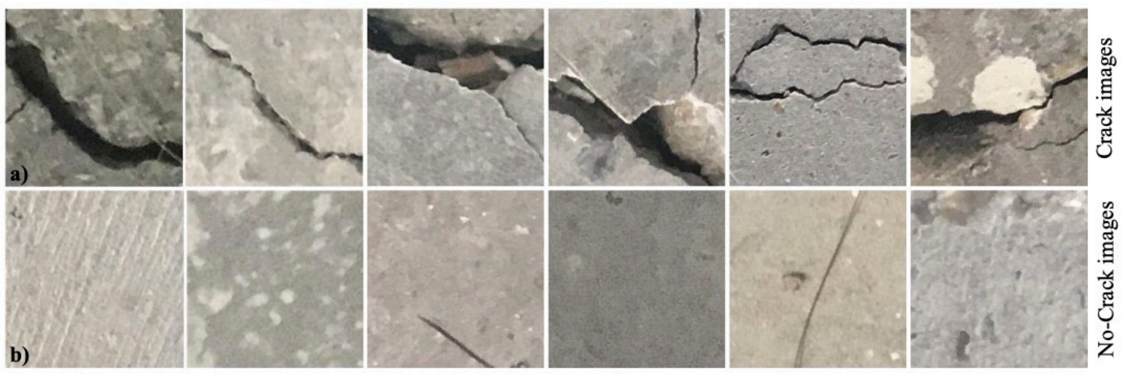


Fig. 3 Images of (**a**) cracked and (**b**) uncracked areas from the collected dataset

2. Data Preprocessing

The following preprocessing techniques were applied to improve the quality of the dataset and enhance the model's performance in crack detection:

Auto-Orient:

- Corrects the orientation of images that might be incorrectly rotated due to camera angles or other factors.
- Ensures that all images are oriented consistently, preventing the model from learning incorrect patterns based on misaligned images.

Resize (Stretch to 256x256):

- Standardizes the size of images to 256x256 pixels to ensure uniform input dimensions for the model.
- Different image sizes can introduce inconsistency in the training process. Resizing the images ensures that the model receives input of a consistent size, improving the learning process and computational efficiency.

RGB to Black and White Conversion:

- Converts RGB images to grayscale (black and white).
- Cracks are visualized clearly in grayscale, and color information is not necessary for detecting cracks. This reduces computational complexity and allows the model to focus on the structural features, such as edges and patterns, without being distracted by irrelevant color data.

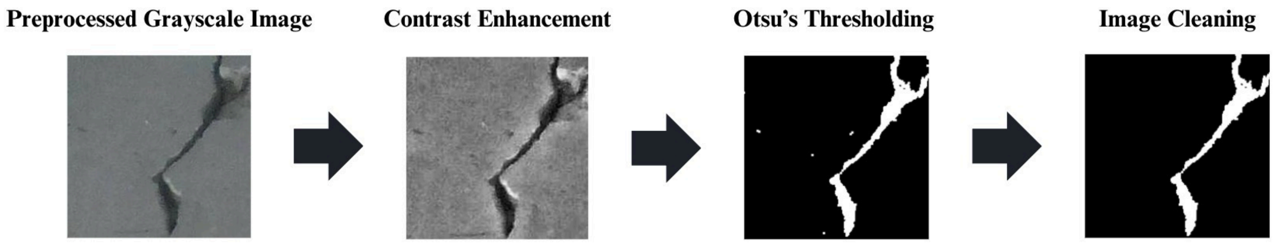


Fig. 4 Image Preprocessing Pipeline

Contrast Enhancement:

- Increases the contrast of the image to make cracks more prominent.
- Cracks can sometimes be faint and hard to detect, especially in low-light conditions. By enhancing contrast, the cracks become more visible, improving the model's ability to detect subtle or thin cracks.

Otsu Thresholding:

- A binarization technique that separates the foreground (cracks) from the background.
- This method automatically determines an optimal threshold value, segmenting cracks from the background even in images with varying lighting conditions. It helps in isolating the crack regions and reduces noise, making the detection task more accurate.

Image Cleaning:

- Removes noise and irrelevant features from the images.
- Images often contain noise or artifacts (e.g., dust, lighting variation) that can confuse the model. Image cleaning techniques like noise removal (e.g., using Gaussian blur) help in removing unwanted elements, ensuring the model focuses on the important features, such as cracks, and performs more accurately.

3. Data Annotation

Data annotation is a process of labelling important information on an image so that it can be used by machines. It is quite useful in supervised machine learning (ML). The system takes labeled datasets as input for processing, understanding, and learning its patterns to yield the desired output. Using the data annotation process, all the cracks present in each image are marked by making bounding boxes as shown in Fig. 5. The object information is used as a reference to assign weights during the training process. For the YOLOv11 algorithm, data annotation was conducted using the software called roboflow, which directly saves labels in YOLO format.

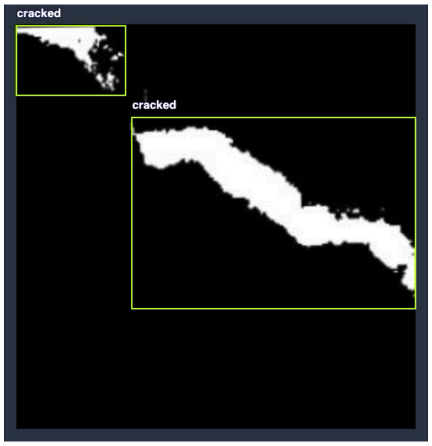


Fig. 5 Annotation of training images for cracked areas

4. Data Augmentation

After data annotation, image augmentation was applied to expand the training dataset and improve model performance. Techniques such as horizontal and vertical flipping, 90° rotations (clockwise, counter-clockwise, upside down), and slight rotations between -10° and +10° were used to simulate different crack orientations. Additionally, random cropping with 10% zoom variations was applied to mimic different perspectives. These augmentations increased the dataset size from 1,887 to 3,287 images, helping the model become more robust and accurate in detecting cracks under various conditions.

5. Model Training

YOLOv11 is an advanced framework of Convolutional Neural Networks (CNN) that excels at tasks like classification, localization, and object detection. In this study, YOLOv11 was utilized for crack detection by identifying changes in color patterns between cracked and uncracked concrete.

Model Development and Training Environment

To develop the YOLOv11 model, a Python environment was used on Google Colaboratory, a free web-based service that offers:

- Maximum Runtime: 12 hours (90 minutes idle time)
- Usable RAM: 12.7 GB
- Disk Space: 107.72 GB
- GPU Support: NVIDIA Tesla T4 with 2,496 CUDA cores & 15 GB of GDDR6 VRAM.

Dataset Splitting

The dataset used in this study was split into three parts for objective evaluation:

- Training Set: 85% (2,800 images, including background images)
- Validation Set: 15% (487 images)

Model Training

Model training involves supplying the training dataset to a machine learning algorithm to minimize a loss function and optimize weights and biases. The following key steps were undertaken:

1. **Transfer Learning:**
   Pre-trained weights from YOLOv11 were used to accelerate training. This drastically reduced training time and improved results.
2. **Model Configuration:**
   a. Framework Version: YOLOv11l (Large variant)
   b. Number of Parameters: 25,280,083
   c. GFLOPs: 86.6
   d. Batch Size: 64

   e. Background Image Percentage: 4% (112 uncracked images added to make a total of 2,912 images)

3. **Training Settings:**
   a. Number of Epochs: 150
   b. Training Time: 1 hour, 14 minutes, and 53 seconds
   c. Optimizer Stripping: Runs saved with reduced optimizer size (51.2 MB for both best.pt and last.pt weights).

6. Performance Testing

To evaluate the performance of the YOLOv11 model, key metrics like precision, recall, F1-score, accuracy, and Intersection over Union (IoU) were calculated. These metrics provide a comprehensive understanding of the model's ability to detect and classify cracks effectively.

- **Precision** measures how many detected cracks are correctly identified.
- **Recall** shows how many actual cracks are detected by the model.
- **F1-score** balances precision and recall, calculated as their harmonic mean.
- **IoU** evaluates how well the predicted bounding boxes overlap with the ground truth, ensuring accurate object localization.

Average precision (AP) represents the area under the precision-recall curve, showing the balance between precision and recall. Mean average precision (mAP) is the average of AP values across different IoU thresholds.

- **mAP@50**: The average precision calculated at a single IoU threshold of 50%. It measures how well the model detects objects with moderate overlap accuracy.
- **mAP@50:95**: The average precision calculated across IoU thresholds from 50% to 95% (in steps of 5%), offering a stricter and more detailed evaluation of detection performance.

Results and Metrics: The YOLOv11l model achieved the following results on the training set

- Precision: **83.9%**
- Recall: **82.5%**
- mAP@50: **91.6%**
- mAP@50-95: **76.7%**

7. Quantifying the crack dimensions

The plan for quantifying crack dimensions focuses on extracting meaningful metrics such as crack area, perimeter, and length by analyzing the processed crack images. Once the preprocessing steps (contrast enhancement, thresholding, and image cleaning) are complete, the cleaned binary image is used to isolate crack regions and calculate their properties.

To achieve this, the cleaned image is analyzed using contours, which represent the boundaries of the detected cracks. Each contour is processed to determine the crack’s area (measured in pixels), perimeter, and length. The length is estimated using the minimum bounding rectangle fitted to the crack. The bounding box helps approximate the crack's orientation and size, providing a visual representation of the crack’s dimensions.

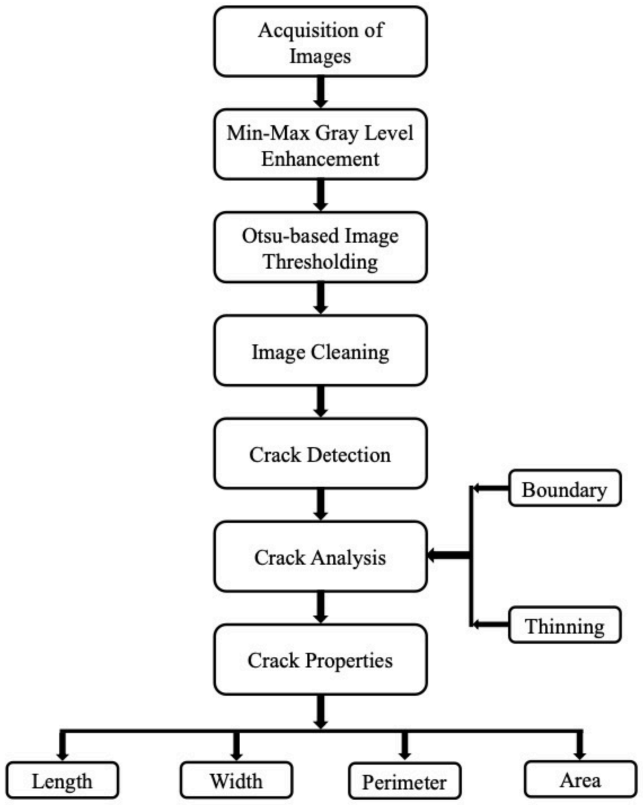


Fig. 6 Flowchart of the process for determining the properties of a crack displayed in an image

For better visualization and verification, the detected cracks and their bounding boxes are superimposed on the cleaned image. This final output not only quantifies the crack dimensions but also provides an interpretable result that highlights the detected cracks along with their measurements. The methodology ensures precise quantification, which can be further used for structural analysis and damage assessment.

Chapter 3

3.1 Crack Detection Results

Experiments

The goal of our experiments was to optimize the crack detection pipeline using YOLO models by identifying the best Background Image %, Batch Size, and YOLO variant. Initially, experiments were conducted on unenhanced images to determine optimal settings. The performance metrics for unenhanced images were not up to the mark, with the best model yielding the following results: YOLOv10m, with a training time of 0.77 hours, achieved a Precision of 0.799, Recall of 0.763, mAP50 of 0.832, and mAP50-95 of 0.685. Subsequently, image quality was enhanced using contrast adjustment and thresholding techniques, which improved crack detection performance. Various YOLO models were tested to find the optimal balance between performance metrics and training efficiency.

Optimization Process

To optimize the Background Image %, YOLOv10n was trained on datasets with varying percentages of uncracked images, ranging from 1% to 10%. After determining the best Background %, YOLOv10n was trained with different batch sizes (64, 32, 16, 8) to find the most effective batch size. Finally, with the optimal Background % and batch size, various YOLOv10 variants (n, s, m, l) were tested to identify the best model for the task. The experiments aimed to maximize Precision, Recall, mAP50, and mAP50-95 while minimizing training time for efficiency

Results from Enhanced Images

With enhanced images, the optimal Background Image % was determined to be 4%, as shown in Table 4. This configuration improved sensitivity to cracks, with the dataset expanded to 2,912 images by adding 112 uncracked images. As seen in Table 5, a batch size of 64 was found to be optimal. Table 6 shows that YOLOv10L was the best-performing model, achieving Precision: 0.821, Recall: 0.808, mAP50: 0.891, and mAP50-95: 0.737. The optimal parameters were chosen by balancing both training time and performance. However, the longer training times led to further exploration of alternative models.

Comparison of Strategies for Improving YOLO Performance

In all the following experiments, we used the optimal settings of 4% background image percentage and a batch size of 64, as determined in previous experiments. Table 4 shows the results of training YOLOv10n for additional epochs. While this improved Recall, it led to slight drops in Precision and mAP due to overfitting. Table 5 demonstrates that increasing the dataset size slightly improved Recall (+2.9%) but caused minor decreases in Precision and mAP (~2-3%) due to noise introduced by augmented images. As seen in Table 6, trying the YOLOv11 model (YOLOv11L) outperformed other models in terms of both metrics and training time.

YOLOv11L showed improvements in Recall (+2.7%) and mAP50 (+1.8%) while maintaining comparable Precision and mAP50-95 to YOLOv10l.

Table 1 Optimal Background Image% for Enhanced Images: YOLOv10n Trained for 50 Epochs *(Best Background Image% = 4)*

Background Image%	**Time (Hrs)**	**Precision**	**Recall**	**mAP50**	**mAP50-95**
1	0.259	0.769	0.718	0.837	0.652
2	0.26	0.792	0.756	0.856	0.659
3	0.269	0.779	0.728	0.837	0.661
4	**0.28**	**0.78**	**0.775**	**0.864**	**0.676**
5	0.282	0.755	0.755	0.841	0.655
6	0.286	0.792	0.785	0.863	0.648
7	0.284	0.767	0.755	0.837	0.619
8	0.288	0.852	0.727	0.869	0.678
9	0.288	0.659	0.728	0.755	0.562
10	0.311	0.782	0.74	0.84	0.645

Table 2 Optimal Batch Size for Enhanced Images: YOLOv10n Trained for 50 Epochs with 4% Background Image *(Best Batch Size = 64)*

Batch Size	**Time (Hrs)**	**Precision**	**Recall**	**mAP50**	**mAP50-95**
64	**0.28**	**0.78**	**0.775**	**0.864**	**0.676**
32	0.34	0.766	0.777	0.865	0.666
16	0.495	0.723	0.775	0.834	0.641

Table 3 Best YOLO Model for Enhanced Images: YOLOv10 Variants Trained for 100 Epochs with 4% Background Image and Batch Size 64 *(Best Model = YOLOv10L)*

Model	**Time (Hrs)**	**Precision**	**Recall**	**mAP50**	**mAP50-95**
YOLOv10n	0.618	0.822	0.741	0.861	0.669
YOLOv10s	0.613	0.798	0.764	0.847	0.664
YOLOv10m	0.753	0.84	0.779	0.883	0.731
YOLOv10L	**1.014**	**0.821**	**0.808**	**0.891**	**0.737**
YOLOv10x	1.356	0.844	0.8	0.883	0.74

Table 4 YOLOv10n Performance Across Different Epochs

Epochs	Time (Hrs)	Precision	Recall	mAP50	mAP50-95
50	0.28	0.78	0.775	0.864	0.676
100	0.618	0.822	0.741	0.861	0.669
200	1.205	0.783	0.775	0.853	0.661

Table 5 Impact of Dataset Size on YOLOv10n Performance with 100 epochs

Metric	2800 Images	3600 Images	Change
Time (Hrs)	0.281	0.345	**23%**
Precision	0.792	0.771	**-2.7%**
Recall	0.756	0.778	**2.9%**
mAP50	0.856	0.846	**-1.2%**
mAP50-95	0.659	0.656	**-0.5%**

Table 6 Performance of YOLOv11L for 100 epochs with Optimal Parameters

Model	Time (Hrs)	Precision	Recall	mAP50	mAP50-95
YOLOv11m	0.778	0.851	0.781	0.901	0.747
YOLOv11L	**0.827**	**0.824**	**0.835**	**0.909**	**0.754**

Table 7 Comparison of YOLOv10L and YOLOv11L Performance for 100 epochs

Model	Time (Hrs)	Precision	Recall	mAP50	mAP50-95
YOLOv10L	1.014	0.821	0.808	0.891	0.737
YOLOv11L	0.827	0.824	0.835	0.909	0.754

Final Results

From the insights provided in the tables, it is clear that YOLOv11L performs the best in terms of both speed and performance. After further training YOLOv11L for 150 epochs, the following validation metrics were achieved:

- Precision: **84.2%**
- Recall: **82.5%**
- mAP@50: **91.6%**
- mAP@50-95: **76.6%**

These results show a significant improvement in detection accuracy and robustness. YOLOv11L achieved higher mAP50 and Precision, along with better Recall and mAP50-95, confirming its effectiveness in crack detection. This makes YOLOv11L the top choice, outperforming previous models and proving its potential for real-world deployment.

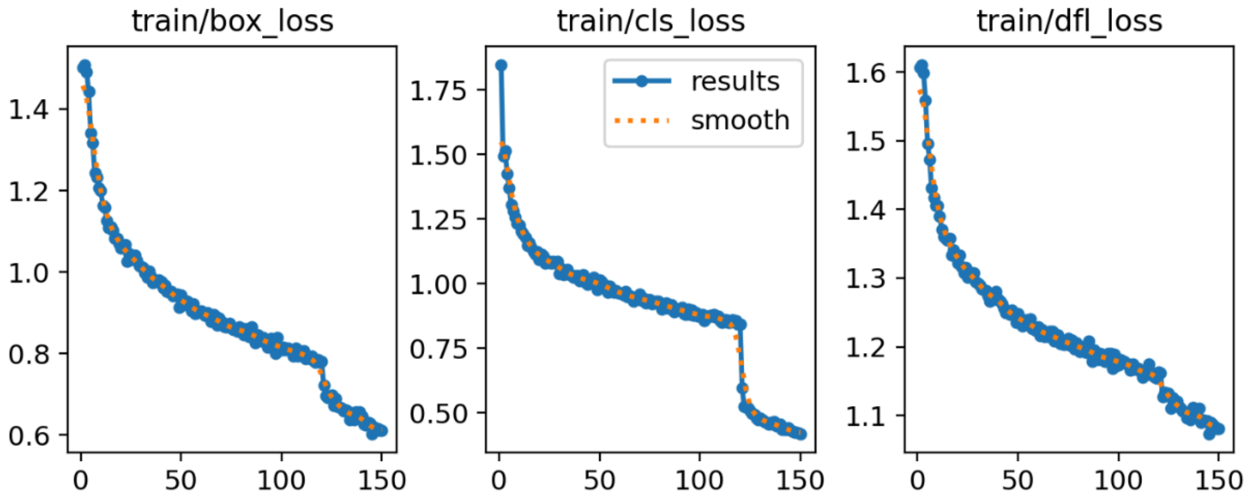
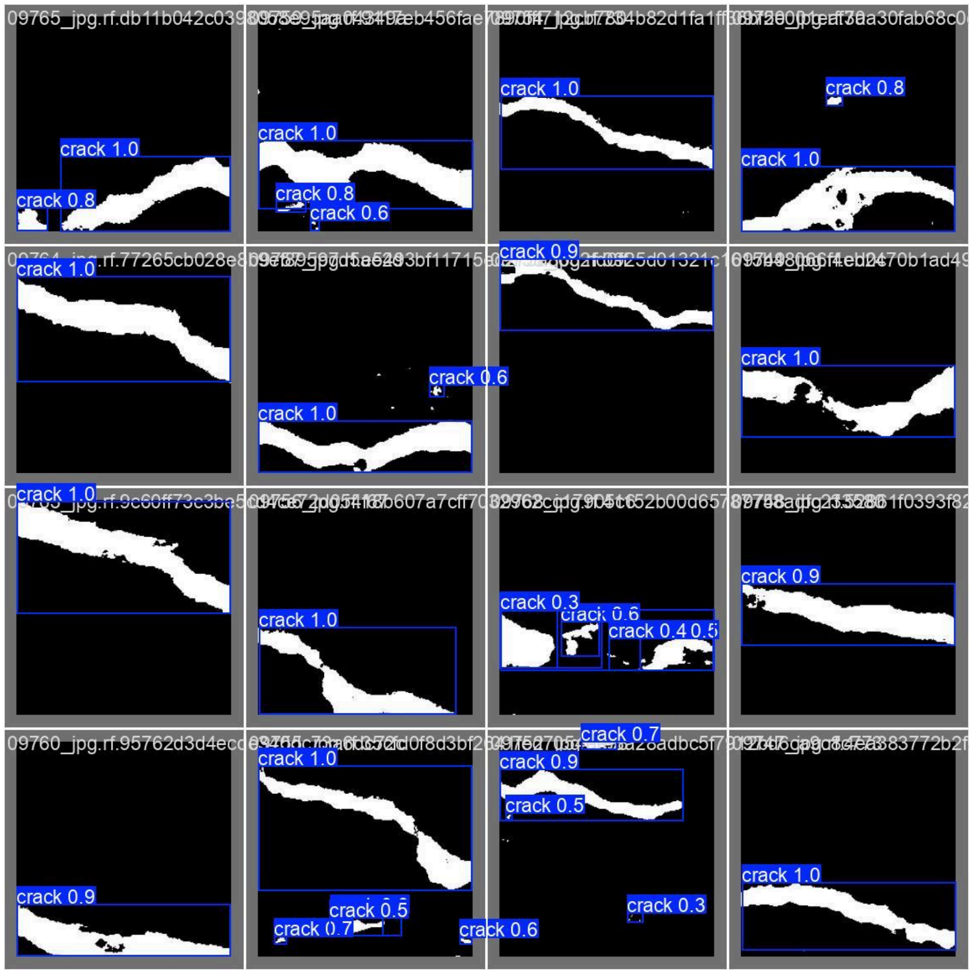


Fig. 7 Plots of loss versus epochs for training dataset of YOLOv11L model

Validation Results



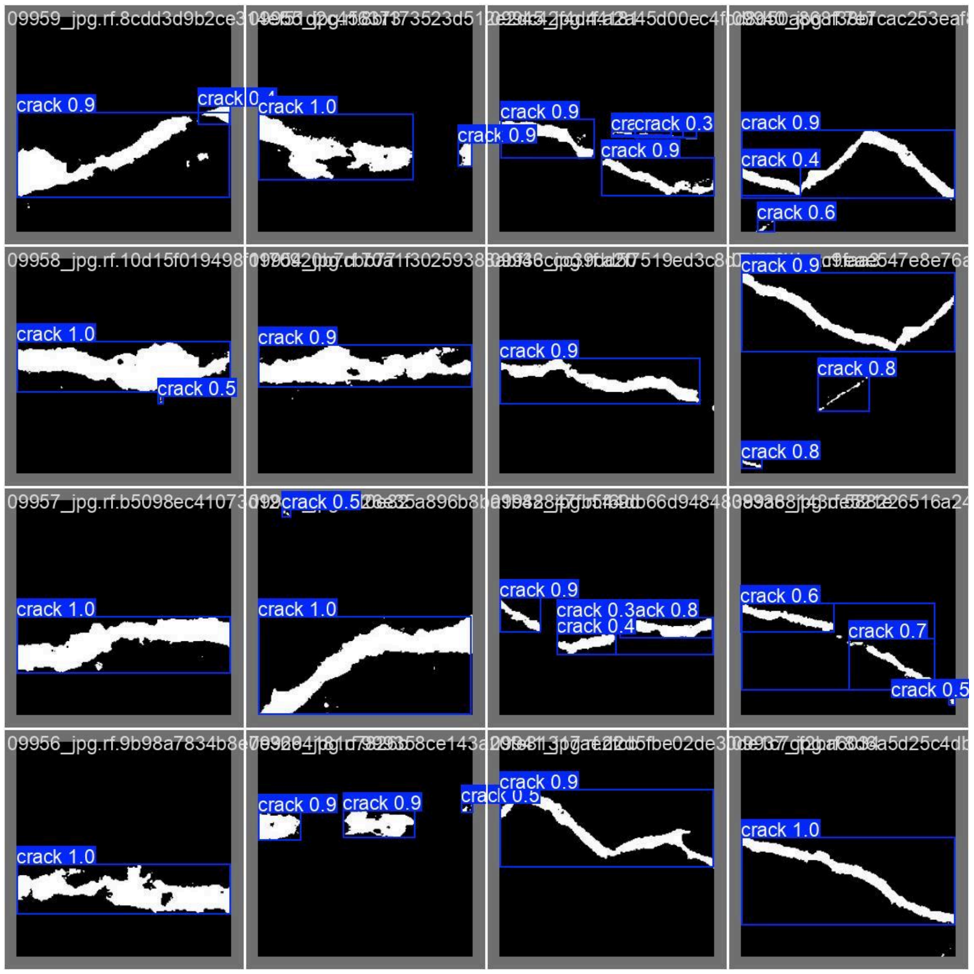


Fig. 8 Crack detection results on the validation set. The images show the predicted crack boundaries overlaid on the original images, with confidence scores indicating the model's certainty. These are predicted labeled images, demonstrating the model’s ability to localize cracks accurately, with varying confidence levels

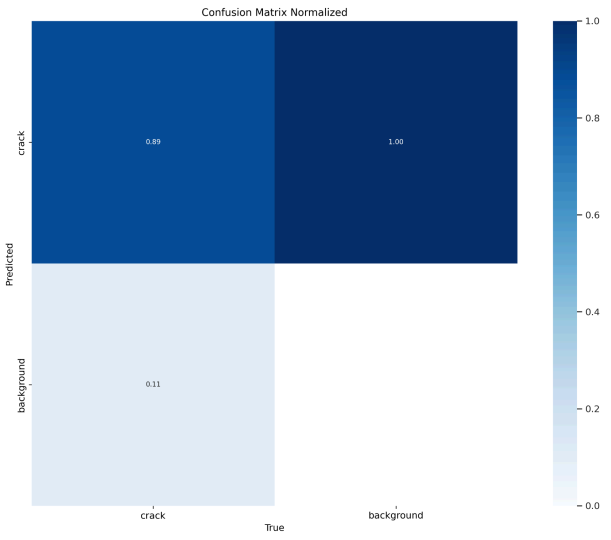


Fig. 9 Confusion matrix of YOLOv11 model

3.2 Crack Quantification Results

The crack quantification process results in key metrics, including area, perimeter, length, and width, derived from analyzing cleaned binary crack images. Each crack is represented as a distinct contour, allowing for precise measurement of these metrics in pixel units. The area is calculated as the number of white pixels within the crack region, representing the total size of the crack. The perimeter is the count of boundary pixels, giving an indication of the crack's edge complexity. The length and width are measured as the major and minor axes of the minimum bounding rectangle fitted around the crack, respectively. This bounding rectangle helps identify the crack's orientation and dimensions.

To convert these pixel-based measurements into real-world units (e.g., meters or centimeters), a calibration step is required. A known reference dimension in the image, such as the size of a calibration object or a specified scale, is used to calculate the conversion factor between pixels and real-world measurements. For example, if a reference object of 10 cm spans 100 pixels in the image, the conversion factor becomes 0.1 cm/pixel. Applying this factor allows all crack dimensions—area, perimeter, length, and width—to be expressed in meaningful real-world units.

This conversion is critical for practical applications such as structural health monitoring, where engineers assess the extent of damage in terms of physical measurements. The ability to translate pixel-based quantifications into real-world dimensions enhances the relevance and utility of the results for evaluating structural safety and planning necessary repairs.

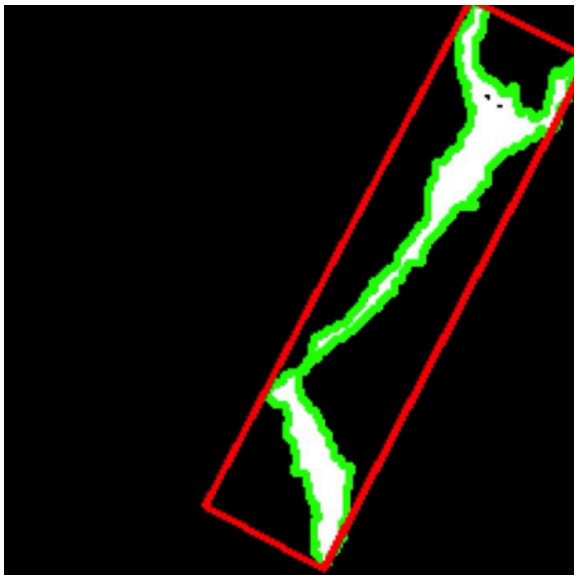


Fig. 10 corresponds to the visual representation of a detected crack and its features. The quantified measurements derived from the figure are as follows:

Crack Area: 2948.50 px^2
Crack Perimeter: 707.03 px
Crack Length: 225.89 px
Crack Width: 53.33 px

Fig. 10 Visualization of Crack Features. The white pixels represent the detected crack region, the green boundary indicates the crack's contour, and the red bounding box highlights the crack's orientation and dimensions.

3.3 Discussion

The results from the crack detection and quantification experiments demonstrate the effectiveness of the optimized pipeline in accurately identifying and measuring cracks.

Crack Detection
Significant improvements were observed after applying image enhancement techniques, such as contrast adjustment and thresholding, which increased sensitivity to crack features. YOLOv11L, trained with optimized parameters (4% background image percentage and batch size of 64), achieved Precision (0.842), Recall (0.825), mAP50 (0.916), and mAP50-95 (0.766) after 150 epochs. The high Recall indicates better crack sensitivity, crucial for detecting small or subtle cracks. However, the slight drop in Precision during longer training suggests the potential for overfitting, necessitating careful monitoring of training duration.

Crack Quantification
The pipeline successfully quantified crack dimensions, such as area, perimeter, length, and width, providing meaningful insights into crack geometry. The conversion of pixel-based measurements into real-world units using calibration enhances its applicability, enabling engineers to assess structural damage with precision. These metrics are vital for damage assessment and repair planning in structural health monitoring.

Real-World Applicability and Challenges
The combined detection and quantification system shows strong potential for real-world deployment. However, challenges such as handling complex textures, maintaining calibration accuracy, and adapting to diverse field conditions remain. Further fine-tuning with domain-specific data and augmentation can enhance model robustness and generalizability.

Future Directions
Future efforts could focus on improving preprocessing techniques, developing methods for detecting micro-cracks, and integrating the system into a broader structural health monitoring framework. Providing a user-friendly interface for calibration and results visualization could further simplify real-world implementation.
In summary, the optimized pipeline achieves reliable crack detection and quantification, marking a significant step toward automated structural health assessment while highlighting areas for ongoing improvement.

Chapter 4

4.1 Conclusion

This study presents a comprehensive approach to crack detection and quantification, leveraging the YOLO object detection framework combined with image preprocessing techniques to enhance detection accuracy. The results demonstrate the effectiveness of optimizing key parameters such as the background image percentage, batch size, and YOLO model variant. Through rigorous experimentation, YOLOv11L emerged as the best-performing model, achieving Precision (0.842), Recall (0.825), mAP50 (0.916), and mAP50-95 (0.766) after training for 150 epochs. These metrics underline the robustness of the optimized pipeline in detecting cracks with high accuracy and minimal false detections.

The quantification stage provides valuable insights into crack dimensions, including area, perimeter, length, and width, derived from cleaned binary crack images. By converting pixel-based measurements to real-world units using a calibration factor, the pipeline becomes highly relevant for practical applications, such as structural health monitoring and damage assessment. This capability to measure cracks precisely enables engineers to assess damage severity, prioritize repairs, and plan maintenance strategies effectively.

Despite its success, the study acknowledges limitations such as the potential for overfitting during extended training, challenges in handling highly textured or noisy images, and the reliance on accurate calibration for real-world deployment. Addressing these limitations through advanced preprocessing techniques, domain-specific data integration, and model adaptation will further improve the pipeline's performance and applicability.

In conclusion, the proposed crack detection and quantification pipeline represents a significant step toward automating structural health monitoring processes. By combining advanced deep learning techniques with precise quantification methods, the system offers a robust and scalable solution for identifying and assessing structural cracks. With further refinements and field validation, this approach has the potential to revolutionize infrastructure management, ensuring the safety and longevity of critical structures.

4.2 References

1. Yuan, J., Ren, Q., Jia, C., Li, M., *et al.* (2023). Automated pixel-level crack detection and quantification using deep convolutional neural networks for structural condition assessment. *Structures*, 59, 105780.
2. Harb, S., Diaz, P. M. A., Maboudi, M., Gerke, M. (2024). Multi-temporal crack segmentation in concrete structure using deep learning approaches.
3. Deng, J., Multani, A. S., Zhou, Y., Lee, V. C. S., *et al.* (2022). Review on computer vision-based crack detection and quantification methodologies for civil structures. *Construction and Building Materials*, 356(1), 129238.
4. Mishra, M., Jain, V., Singh, S. K., Maity, D. (2022). Two-stage method based on the you only look once framework and image segmentation for crack detection in concrete structures. *Architecture Structures and Construction*.
5. Mishra, M., Barman, T., Ramana, G. V. (2022). Artificial intelligence-based visual inspection system for structural health monitoring of cultural heritage. *Springer-Verlag GmbH Germany*, Published online: 31 October 2022.
6. Mishra, M., Lourenço, P. B., Ramana, G. V. (2023). Deep learning-based YOLO network model for detecting surface cracks during structural health monitoring. *Springer-Verlag GmbH Germany*.
7. Mishra, M., Lourenço, P. B. (2024). Artificial intelligence-assisted visual inspection for cultural heritage: State-of-the-art review. *University of Minho, ISISE, ARISE, Department of Civil Engineering*.
8. Karimi, N., Mishra, M., Lourenço, P. B. (2024). Automated surface crack detection in historical constructions with various materials using deep learning-based YOLO network. *International Journal of Architectural Heritage*.