DESIGN AND DEVELOPMENT OF CONCRETE SURFACE CRACK ANALYSIS SYSTEM

PROJECT REPORT

Submitted by

VIDYA VIJAYAN

REG. NO.: TKM20CESC18

ROLL NO.: M20CESC18

Department of Civil Engineering

T.K.M. College of Engineering, Kollam- 691005

A. P. J. ABDUL KALAM TECHNOLOGICAL UNIVERSITY

2022

Thangal Kunju Musaliar College of Engineering, Kollam, Kerala.

DEPARTMENT OF CIVIL ENGINEERING

CERTIFICATE

This is to certify that this report entitled

'DESIGN AND DEVELOPMENT OF CONCRETE SURFACE CRACK ANALYSIS SYSTEM'

is presented by

VIDYA VIJAYAN

in the year 2022 in partial fulfilment of the requirements for the award of the Degree of Master of Technology in Civil Engineering by the

A.P.J. Abdul Kalam Technological University.

Guide	*Co-ordinator*	*Head of the Department*
Dr. Chinsu Mereena Joy	**Dr. Ramaswamy K. P**	**Dr. Sajeeb R**
Assistant Professor	Professor	Professor
Department of Civil Engg.	Department of Civil Engg.	Department of Civil Engg.
T.K.M.C.E., Kollam	T.K.M.C.E., Kollam	T.K.M.C.E., Kollam

ACKNOWLEDGEMENT

I take this opportunity to express my deep sense of gratitude and sincere thanks to all who helped me to complete the project successfully.

I am deeply indebted to my guide, **Dr. Chinsu Mereena Joy,** Assistant Professor, Department of Civil Engineering for her excellent guidance, positive criticism, and valuable comments.

I am greatly thankful to my project coordinators, **Dr. Ramaswamy K. P,** Assistant Professor, Department of Civil Engineering for their constant supervision as well as for providing necessary information regarding the project.

I am greatly thankful to **Dr. Sajeeb R**, Professor and Head of the Department of Civil Engineering, for his kind support.

Finally, I thank my parents and friends who directly and indirectly contributed to the successful completion of my project.

VIDYA VIJAYAN

ABSTRACT

Structural Health Monitoring (SHM) refers to a periodical inspection to monitor the condition and characteristics of civil structures. On the other hand, damage can be interpreted as a structural modification that changes physical properties and weakens the structure, so it should be addressed as early as possible to prevent additional damage. Usually, cracks are visually monitored by inspectors who record data regarding presence, location, and width. Manual visual inspection is often deemed ineffective in terms of cost, protection, the accuracy of assessment, and reliability. As moving with the fast pace of technology advancements, the possibility of the information technology-driven methodologies in the construction field is also getting wide visibility. In surface crack, detection different technology-backed automated systems outperform the traditional manual inspection and crack detection. With the help of different computational aids like Image Processing, Machine Learning, and Deep Learning techniques, the images, and videos captured from surveillance sites are analyzed for automated crack detection.

In this study design and development of a concrete surface crack analysis system was done using image processing machine learning and deep learning techniques. The major components include an acquisition module, an analysis module, and a client application module. The acquisition module is designed and developed as an IoT device that can be controlled over the internet using a mobile application, and it is intended to move over the surface and capture the cracks. The analysis module is integrated with the web server, build using python programming and flask library. The core analysis process was implemented using computer vision and deep learning algorithms. The classification of crack images to cracked or non-crack was done by convolutional neural networks, the segmentation and localization are carried out with image processing. The client applications have two versions, one is a web app and another is a mobile app that is used to control the accusation and to interact with analysis. The system is validated with a set of crack and non-crack images of beams and walls collected from the concrete technology lab, TKM.

Keywords: Concrete surface cracks, SHM, Machine learning, Deep learning

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 General

The term "Structural Health Monitoring" (SHM) describes a periodical inspection to keep track of the state and characteristics of civil structures. The SHM system is used to track the condition of the structure in comparison to normal circumstances. Its major objective is to determine the changes in the civil structure, create a maintenance plan, and implement the necessary measures to address the changes in the structure. But there are numerous ways to keep track of structural health using various structure health indices. Damages are structural modification that alters the physical characteristics and makes it weaker. The importance of concrete cracks should be increased in terms of structural harm and durability. These days, many developed nations regularly evaluate structures for maintenance. In general, a crack is a flaw that could lead to catastrophic consequences and damage to the structure. Cracks are one of the important aspects that determine the structural condition for the safety and maintenance of structures like highways, subways, bridges, buildings, dams, tunnels, monuments, etc., and they should be found as soon as feasible.

Therefore, a crucial component of damage evaluation is the detection of cracks. The visual inspection method is the widely used method to collect the crack details such as presence, position, and width, and these details canbe used for maintenance purposes Even though it is the most widely used method, it has numerous drawbacks like expensive, time-consuming, labor-intensive, and sometimes inaccurate because of the lack of expertise. However, for large infrastructures, visual inspections are rather time-consuming and difficult, let alone the safety aspects when dealing with areas that are hard to reach. Unmanned aerial vehicles (UAVs) are used by several companies to accomplish this, particularly because the UAV market offers dependable, user-friendly, and reasonably priced UAVs that can boost inspectors' performance. Although commercial UAVs are getting simpler to use, this does not necessarily mean that the workflow for damage assessment from data gathering to assessment is also simpler. Since UAV allows for the acquisition of a greater dataset than that through conventional visual examination employing cameras, the latter poses the most barrier. The time saved by collecting data from a structure can easily be used for data analysis since deriving useful insights from such a large volume of data can be time-consuming and labor-intensive. To improve the

workflow of UAV-based inspection, new paradigms for the evaluation of concrete structures, such as those that automatically detect concrete fractures, are needed. This will result in a safer, less expensive, and more objective analysis. Although the creation of an automatic crack detection system is not wholly new, it is still a complex task that has been researched over the past few decades.

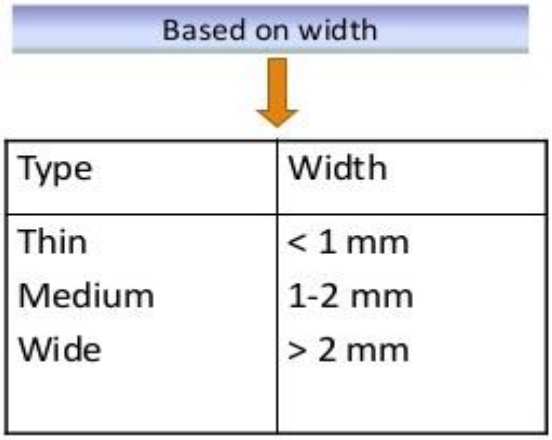
1.2 Crack

Concrete cracks refer to a complete or incomplete separation of either concrete or masonry into two or more parts produced by breaking or fracturing (ACI Concrete Terminology). Based on occurrence cracks can be categorized in either plastic concrete or hardened concrete. Appearance, severe structural distress, or lack of durability are the various factors that are affected by the formation of the concrete crack[7]. The overall extent of the damage may be reflected by cracks, or they may point to issues of greater severity.

Cracks can be classified based on direction, width, depth, and when they occur as shown in fig.1.2.7.

1.2.1 Based on the width of the crack

Table 1.1: classification of cracks based on width



1.2.2 Based on the sight of a crack

Vertical cracks are caused by shrinkage or thermal and are typically low risk in poured concrete foundations that seem to appear almost straight or wandering, usually even in width, intermittent, or most often straight. Settlement is frequently the cause of diagonal cracks that appear at the maximum height of the foundation wall. A diagonal crack may also be caused by this settling if the footing on one side of the wall is experiencing settlement problems. The most likely cause of horizontal cracks in the center of the wall

is an applied load, such as poorly or prematurely compacted backfill around the base, soil compacting during settlement, hydrostatic pressure from a high water table against the foundation, a lack of drainage against the foundation wall, or heavy machinery operating too nearer to the foundation wall. Horizontal cracks discovered high up on the wall are most frequently the result of frost damage.

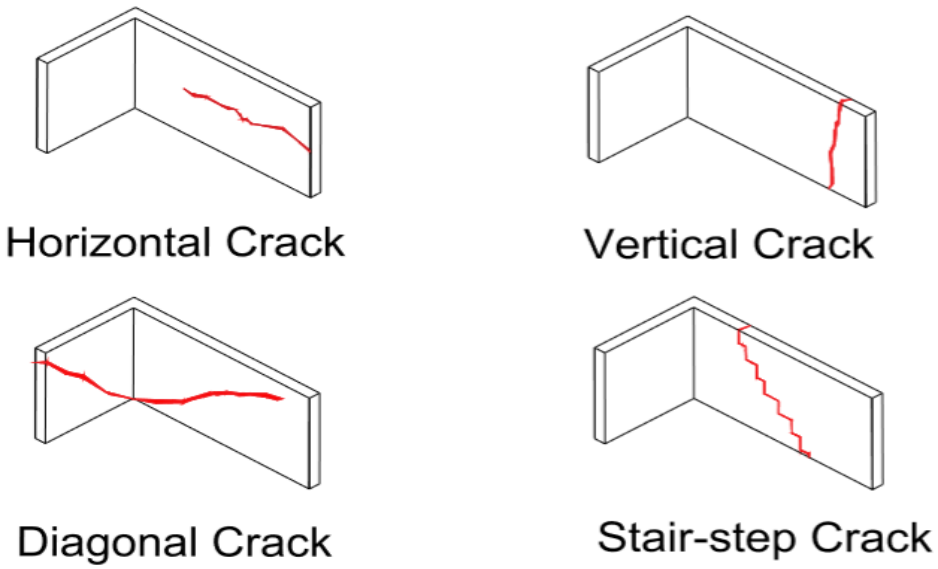


Figure 1.1: classification of cracks based on sight

(Source: Vijayan et al., 2021)

1.2.3 Plastic concrete cracks (non-structural cracks)

- Plastic shrinkage cracks

As moisture from freshly laid concrete evaporates quicker than it is replaced by bleeding water, the surface concrete contracts.

- Settlement cracking

The underlying ground's structural weakness will lead to the settling of cracks. For example, its roots decay when a large tree is removed and leaves large voids behind which soften the ground.

- Cracks due to framework movement

The movement of the formwork after the concrete has started to harden will create cracks before it has grown strong enough to support its weight.

1.2.4 Hardened concrete cracks

- Drying shrinkage cracks

The cracks due to drying shrinkage are is occurred due to the moisture loss from the concrete, which can shrink about 1 percent.

- Cracks due to thermal stresses

Temperature variations within concrete structures may result from different areas of the structure releasing hydration heat at different rates or from one part of the structure cooling or heating more quickly than another part of the structure does depend on the weather.

- Cracks due to chemical reactions

Figure 1.2: Plastic shrinkage cracks and settlement cracks

(Source: Vijayan et al., 2021)

- Weathering cracks

Weathering processes such as freezing and thawing, wetting and drying, and heating and cooling can cause cracks in the concrete structure.

- Cracks due to poor construction practices
- Cracks due to corrosion of reinforcement

Concrete cracks eventually form as a result of the electrochemical process of metal corrosion, which involves the movement of oxidizing chemicals, humidity, and electrons inside the metal. Various chemical reactions also occur on and around the metal surface.

- Cracks due to construction overloads
- Cracks due to design and detailing errors

The consequences of improper design or detailing vary from poor appearance to lack of serviceability to devastating failure.

- Cracks due to externally applied loads

Load-induced tensile stresses are major causes of cracks in concrete.

1.2.5 Structural cracks

- Flexural cracks

In the tensile zone, cracking in reinforced concrete flexural members subjected to bending begins in the tensile zone, e.g. at the beam soffit.

- Shear cracks

These are triggered after the concrete has hardened by structural loading or motion. Shear cracks due to the combined effects of bending and shearing action are best characterized as diagonal stress cracks. Beams and columns are normally susceptible to cracking like that.

- Internal micro-cracks

Due to high differential cooling rates, or due to compressive loading, micro cracking may occur in severe stress zones.

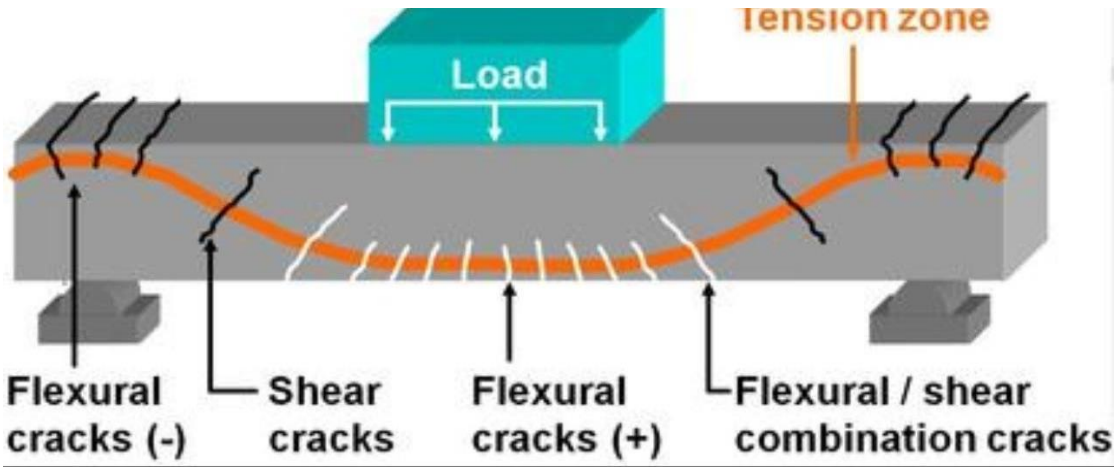


Figure 1.3: Flexural and shear cracks

(Source: Ali et al., 2021)

1.2.6 premature drying cracks

Premature drying can result in two common types of cracks.

Surface cracks known as crazing cracks, which resemble broken glass or spider webs, are very small. When the surface of a concrete slab loses moisture too quickly, crazing cracks are likely to develop. Crazy fractures are not a structural issue, despite being unattractive. Crusting cracks commonly appear during the concrete stamping process, which is a means to give concrete surfaces structure or pattern. On bright or windy days when the top of the slab dries out faster than the bottom, the concrete surface may get crusty.

Figure 1.4: Cracks due to premature drying

(Source: Ali et al., 2021)

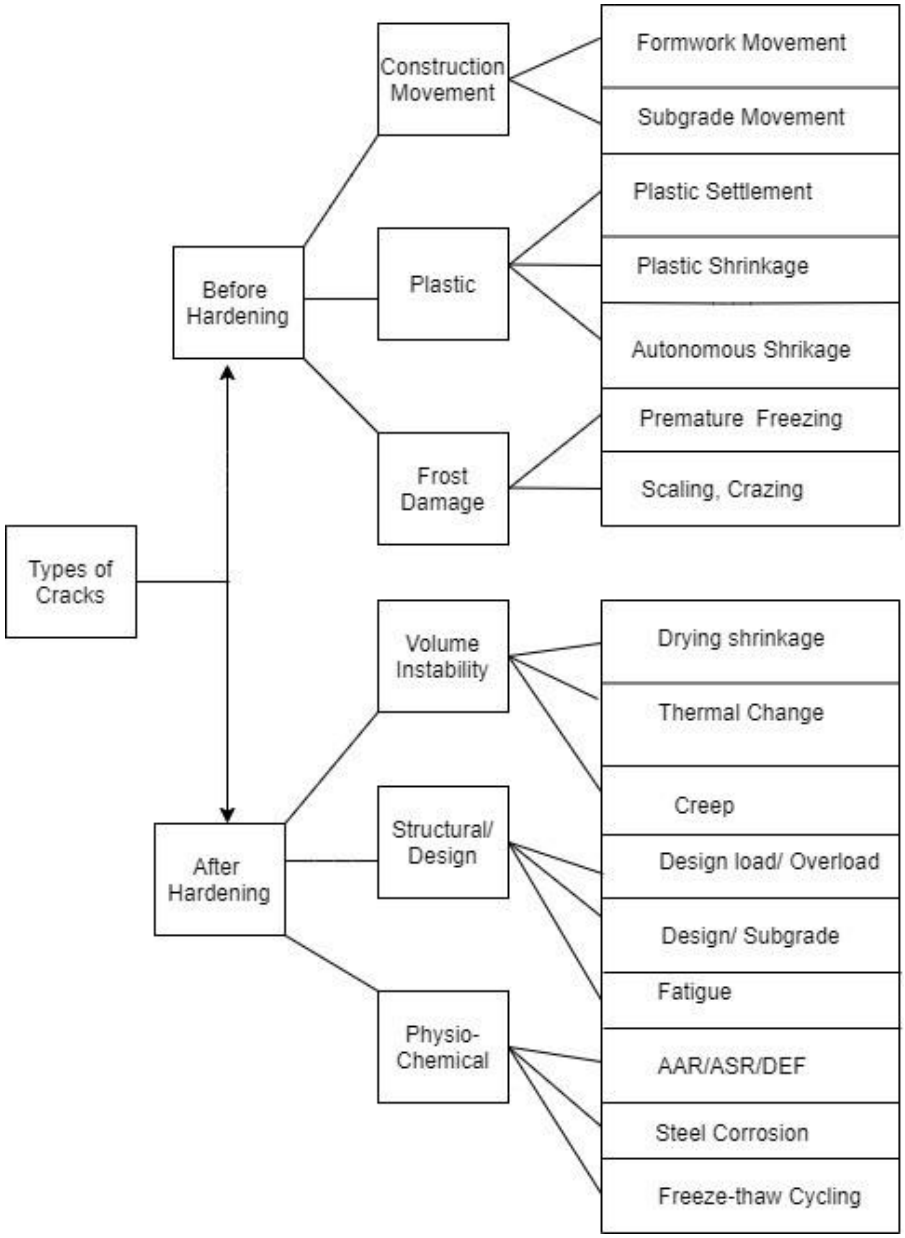


Figure 1.5: Types of cracks

(Source: Ali et al., 2021)

1.3 EVALUATION OF CRACKS

When the repair of cracks in concrete is expected, it is necessary to first define the position and scope of cracking. It is done to check whether it is active or dormant, presence of reinforcement and also to find the width of crack. Cracks need to be fixed if the strength is decreased, rigidity or toughness of the structure to an undesirable structure stage, or if the structure's role is extremely affected. In certain situations, such as cracking in structures that hold water, the structure's purpose would be dictate the need for repair, even if strength, rigidity ,appearance is not affected greatly[29].

Direct and indirect observations, nondestructive and destructive testing, and tests of cores taken from the structure are the various methods to find the information on the general condition of concrete in a structure, location and extent of cracking etc. Drawings, construction and maintenance records can also be used for the same.

1.3.1 Direct and indirect observations

A high quality microscope is used for direct observation of crack widths in concrete structures. The apparatus operates by an adjustable lamp unit and the image is focused by turning a knob. The eyepiece scale can be turned through 360° to align with the direction of the crack or pitch under examination to an accuracy of approximately 0.025 mm as shown below in fig 1.6.

Figure 1.6: Crack width microscope for measuring crack width

Crack width ruler is an alternative to a crack microscope for estimating crack thicknesses. It is similar in size to a standard credit card marked with a range of graded lines as shown in fig 1.7.

The mechanical crack meter is used to monitor opening & closing of the surface crack in bricks, cement, or masonry works. It consists of two plates that partly overlap. The upper plate has an engraved cross label, while the lower plate has a grid divided into millimeters,horizontally and vertically. The ends are sealed with adhesive or plugs over

the crack to check whether cross mark coincides with the zero mark of the Cartesian axis of the grid.

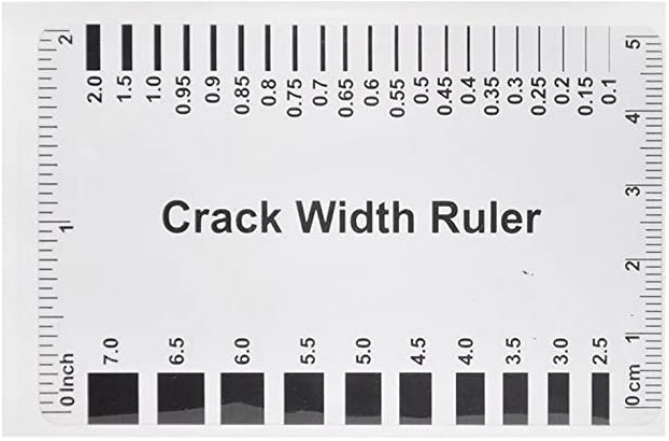


Figure 1.7: Crack width ruler for measuring crack width

Figure 1.8: Crack monitor for measuring crack width

1.3.2 Non-destructive testing

Nondestructive testing is the way to estimate the presence of internal cracks, voids and the penetration depth of concrete cracks. Tapping with a surface hammer is an easy way of recognizing laminar cracking close to about the surface. One or more cracks below are specified by a hollow sound and being hammered parallel to the surface.

Pachometer is to evaluate the presence of reinforcement bars. The existence of steel bars is shown by Pachometers and allows the experienced person to determine reinforcement depth and width However, it is required to remove some portion of cover concrete to pinpoint the bar sizes and cover in reinforcement congested areas. Results obtained from Pachometers are analyzed by the use of computer algorithms and magnetic fields. This will provide a visual picture of the reinforcing bars layout in the scanned area. Pachometers is very useful in detecting reinforcement bars, estimate the position and reinforcement size and measure concrete cover.

Concrete cracks can be detected by the use of an ultrasonic non-destructive test equipment. Propagation of mechanical wave from one face of the concrete member to opposite face is taking place. The time taken by wave to travel through the member is measured by means of some electronic device. Pulse velocity is then evaluated with the distance between the transmitting and receiving transducers. If a wave takes more time to travel from transducer to receiver, it is said that the segment is cracked. Greater the wave velocity suggests the high quality of the concrete. To strengthen the understanding of the outcome, use of an oscilloscope that offers a visual presentation of the signal is preferred. Interpretation of result is difficult in the case of fully flooded crack section hence this instrument is not used.

1.3.3 Tests on concrete cores

Concrete cores provide required crack information which are taken in various roles. It also provides right data on crack thickness and width. Compressive strength test of concrete will not consider the cracked core of concrete specimen. Cracked concrete photographic test results will tell us material that causes cracking, relative paste ratio w/c, age of cracks, secondary deposits on fracture surfaces, the volume and distribution of components of concrete.

1.4 Problem Statement

This project aims to structurally and functionally analyze the different types of surface cracks that appear in the concrete structures and thereby find out the impact on structural health; also to develop an automated surface concrete crack identification system incorporating advanced technologies to effectively identify the type and width of the surface cracks.

CHAPTER 2

LITERATURE REVIEW

To overcome the drawbacks of conventional human inspection procedures, automatic crack detection models were developed. For non-destructive testing, automatic crack detection is very successful. It is hard to test degradation objectively through manual inspection. Fully Automated crack detection can be implemented using various computational methods, such as image processing, machine learning, and combining these two. In this section, such computer-aided methods are elaborated for the convenience of these techniques and are classified into image processing and machine learning methods.

2.1 Image Processing Approaches

Using image processing, different information regarding the cracks, such as presence, localization, and dimensions, can be determined from surface images of the concrete structures. The core image processing techniques like binarization, edge detection, and mathematical morphology are efficiently used for crack detection[4]. Image binarization is an operation to convert a greyscale image to a binary image[20]. According to a threshold, the pixels having a greater value than a threshold are converting to white, and the pixels with lesser value are converting to black or vice-versa. This operation is working perfectly for crack images since cracks appear darker in lighter backgrounds. Edge detection can be used to localize the cracks and further analysis by modifying the properties of cracks present in the images for better identification; different morphological operations can be used[6],[19].

The first step in image processing is image acquisition, to transform the analog image into digital by sampling and quantization processes[5]. Next, to eliminate noise and deterioration in a concrete structure image, pre-processing was performed[10]. Even though these are the general steps followed in crack detection using image processing, there are some other relevant combinations of these techniques discussed in detail as follows.

Lins and Givigi (2016)[17]proposed a model to automate crack detection using the machine vision concept as shown in fig 2.1. A series of images are processed, and the

crack measurements are measured using this technique with only a single camera mounted or even in a robot. According to the proposed method, by using a crack detection algorithm, a series of images were analyzed. The proposed model outputs an image along the observed crack. In a vector that is passed along to the crack calculation algorithm, the pixel locations of the particles are stored. The number of pixels in a cross-section is calculated by the algorithm and outputs the crack dimension.

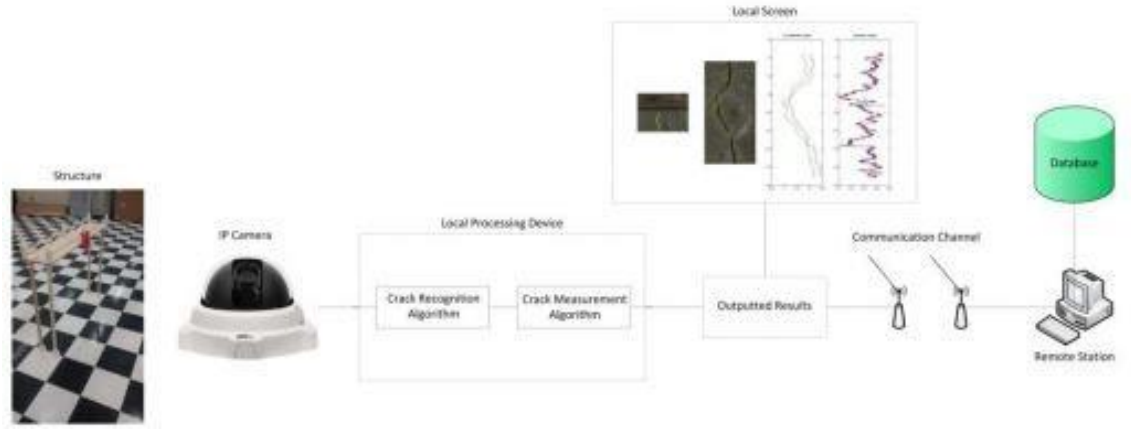


Figure 2.1: Flow diagram of the crack measurement system proposed
(Source: Lins and Givigi (2016))

A system for automated cracking has been suggested by Yusuke and Hamamoto (2011)[28] for the identification of noisy concrete surface images. They aimed to find the length of the cracks. The system contains two Pre-processing steps and two identification steps. Original images were used for the pre-processing step. They have used a real dataset. The accuracy was about 90-95 percent. The limitation of this work was that they had evaluated robustness and accuracy quantitatively by using 60 actual noisy concrete surface images.

Shan (2015)[11] proposed an image processing technique by projecting a crack edge curve on the concrete surface to 3D coordinates; thereby, crack width is assessed by the minimum distance between two sides of the crack edge.

Yamaguchi (2010)[26] uses percolation-based image processing with reduced processing time using the termination and skip-added procedure.

Su (2013)[24] utilized several computer vision methods on CCD images to analyze and crack detection in the concrete structure.

Hoang (2018)[22] studied about detection of surface crack in building structures using image processing technique with an improved otsu method for image thresholding.

2.2 Machine Learning Approaches

Jitendra (2020)[21] proposes a classification model using extracted functions or feature descriptors and different classification algorithms.

Yokoyama and Matsumoto (2017)[25] developed an automatic detector of cracks in concrete using machine learning techniques.

2.3 Deep Learning Approaches

The effects of various factors, such as training data quantity, data heterogeneity, network complexity, and the number of epochs, were also examined by Ali et al.(2021)[1] in their development of a tailored convolutional neural network for concrete surface crack detection. They examined the customized CNN model, the VGG-16, VGG-19, ResNet-50, and Inception V3 models, among others. The CNN and VGG-16 models outperform the other four in terms of accuracy, and they also concluded that over-fitting is caused by increasing the number of training data while decreasing the variety of the data set.

Ali et al. (2022)[2] give a review on deep learning architecture ie, CNN. The main conclusions are as follows.

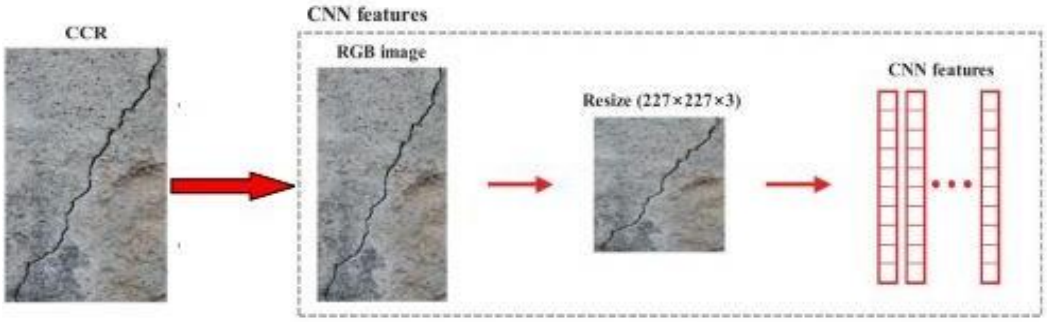
1. Accuracy, Precision, Recall, and F1-Score are the most often used matrices for crack detection evaluation.

2. Fine crack segmentation is well suited to encoder-decoder designs like U-Net, Seg-Net, and FCN architectures.

Chen et al. (2019) [3] developed an automatically detecting algorithm that is capable of detecting whether an image contains cracks and marking the location of the cracks. Adam optimization algorithm and batch normalization (BN) algorithm were used and got an accuracy of 99.71%.

Convolutional neural networks and transfer learning were utilized by Feng et al. (2019)[6] to detect concrete surface cracks in an image taken from a hydro-junction facility using a high-definition camera. To extract image features, the Inception-v3 network is used and achieved a final test accuracy of 96.8%.

Kim et al. (2020)[15] aim with a machine learning framework for the classification of crack images based on the crack candidate regions (CCRs) (i.e. shape and color). After performing binarization for feature extraction on crack and non-crack images, CCRs are extracted. CCRs were generated and then pipelined with the SURF-based and CNN-based methods to construct classification models. Global features are directly used for the classification. The classification model is trained using a collection of images, where the



input images are divided into a group of fixed resolution sub-images. Lastly, these CNN features are passed through the output layer for label categorization.

Figure 2.2: Feature extraction process of CNN
(Source: Kim et al. (2019))

Gang et al. (2018)[27] proposed an algorithm based on deep learning techniques for concrete surface crack detection. Images for the detection were taken from various completed construction sites using a mobile camera, and the main conclusions of their studies are as follows.

1) Convolutional neural network structures are used for image recognition purposes as they overcome real environmental interference such as light and distance.

2) This CNN-based model shows an accuracy of 93.7%.

Silva and Lucena (2018)[23] developed a crack recognition tool based on machine learning using a convolutional neural network Crack detecting process framework (CNN). The trained dataset used is a collection of 3500 images. A crack detection model with an accuracy of 92.27% for the data set was developed.

Li and Zhao (2019)[16] proposed a method for crack detection from images using a convolutional neural network. All the images used for the training are captured from real concrete surfaces using a mobile phone. There are over 1250 captured images, and these images were again sliced into 256 x 256 sizes to obtain a training and validation dataset

of 60000 images. After the training phase of the CNN, to validate the correctness of the trained model a test set of size 205 was applied and obtained an accuracy of 99.09%.

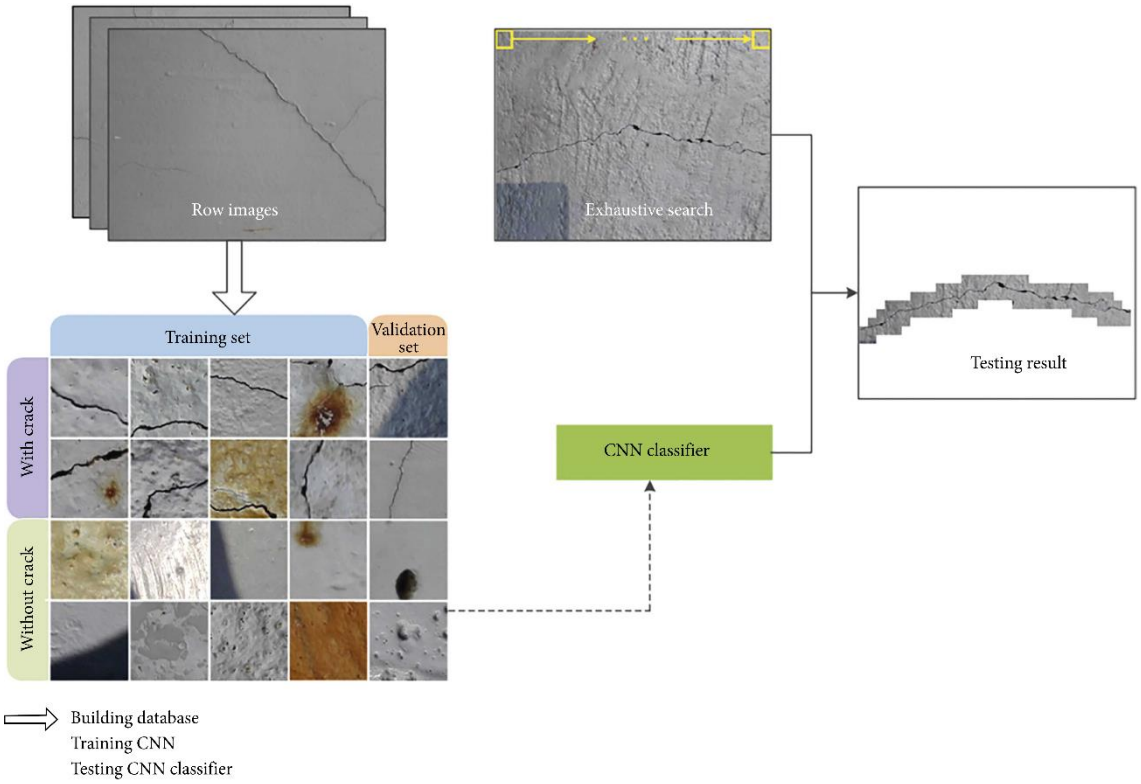


Figure 2.3: Flow chart for detecting cracks using a CNN

(Source: Li and Zhao (2019))

Islam (2019)[9], Mandal (2018)[18], Krizhevsky (2012)[30] and Yao (2018)[27], applied convolutional neural networks and their variants for the automated detection and identification of surface cracks. To make it more useful the trained CNN model is integrated into a smartphone application named Crack Detector. The proposed method is capable of detecting the cracks on real concrete surfaces without any noise.

2.4 Summary of the Literature

From the review of literature, it is clear that the traditional crack detection methods i.e. manual inspection take lots of time and experienced effort, and also it may not give accurate results. Apart from that, the computer-aided methods are showing exceptional efficiency in concrete crack detection. The technologies like image processing, machine learning, and deep learning techniques are well utilized for crack detection. The methods relying on deep learning techniques are the state of art methods that shows astonishing results, about 95% efficiency.

CHAPTER 3
OBJECTIVES AND SCOPE

3.1 Gaps Identified

The research gaps identified from the literature survey are as follows: -

1. Apart from the mere development of algorithms, an end-to-end system is not available
2. Portable data acquisition equipment using modern IoT (Internet of Things) systems was not developed
3. Computational frameworks for crack analysis using image processing, computer vision, and deep learning algorithms
4. Report generation and visualization portal

3.2 Objectives of the Study

The objectives of the study are as shown below:

1. Develop an end-to-end system for structural crack analysis

   This can be achieved through:

   a. Develop a portable acquisition module.
   b. A communication module and a server.
   c. Develop a crack classification module
   d. Develop a crack localization module
   e. Develop a crack segmentation module.
   f. Develop a result analyzer and report generation module
2. Find out the type and width of the detected crack

3.3 Scope of the Study

The scope of the study is to

1. Develop an end-to-end system for structural crack analysis
2. Investigate the hardware technologies need for developing such a system
3. Identify computational methods for automating the crack analysis process
4. Find out the different possibilities in automatic crack analysis through image processing, computer vision, machine learning, and deep learning.

CHAPTER 4
PRELIMINARY

4.1 Image Processing

Image processing is a technique or procedure applied to an image to improve it or find some important information from it. We may also state that the use of computer algorithms, improves or find some relevant information from an image. It is a form of signal processing in which a picture serves as the input, and the output may be another image, features, or characteristics associated with the input image. The basic steps involved in Image processing are:

- Importing the image using tools for image acquisition
- Analyzing and altering the image
- The final result can be an altered image or image analysis report.

4.1.1 Image

An image is defined as a two-dimensional function, F(x,y), where x and y are spatial coordinates. The amplitude of F at each given pair of coordinates (x,y) is referred to as the intensity of that image at that location. We refer to it as a digital image when F's x, y, and amplitude values are all finite. In other words, a 2D array set up in rows and columns can be used to define an image.
Each element in a digital image has a specific value at a specific position and is built using a finite number of elements. These components are also known as pixels, image components, and picture components. The term pixel refers to the basic unit of a digital image.

4.1.2 Types of an image

1. Binary Image– The binary image merely consists of two pixels, 0 and 1, where 0 denotes black and 1 denotes white. Monochrome is another name for this image.
2. Black And White Image– The image contains only black and white colors

3. 8-Bit Color Format– The most popular image format is this one. A grayscale image contains 256 different color tones. In this format, 127 denotes grey, 255 is white, and 0 denotes black.
4. 16-Bit Color Format– It's a format for colored images. It comes in 65,536 distinct colors. High Color Format is another name for it. Compared to a grayscale image, the distribution of color is not the same.

Red, Green, and Blue are the other three formats that make up a 16-bit format. the well-known RGB format.

4.2 Machine Learning (ML)

Machine learning (ML) focuses on the learning component of artificial intelligence (AI) by creating algorithms that best reflect a set of data. Unlike traditional programming, where a given set of features can be used to explicitly write an algorithm. To create an algorithm that may employ innovative or distinct combinations of characteristics and weights that can be generated from the fundamentals, machine learning uses subsets of data. supervised, unsupervised, semi-supervised, and reinforcement learning are the four most popular learning techniques in machine learning, and each is effective for tackling a distinct problem.

(A) In traditional programming, an algorithm and dataset are given to the computer. The computer receives instructions from the algorithm on how to process the dataset and produce outputs.

(B) In machine learning, a dataset and related outputs are given to a computer. As it learns, the computer creates an algorithm that explains how the two are related. Future datasets can be inferred using this approach.

In ML, there are two types of data — labeled data and unlabelled data.

(A) Labeled data involves a lot of human labour to label the data in the first place but includes input and output parameters in a completely machine-readable form.

(B) Data that aren't labeled only have one or no parameters in a machine-readable manner. This eliminates the need for human work but calls for more difficult fixes.

There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods are used today.

4.2.1 Types of Machine learning

a. Supervised Learning:

In supervised learning, have both raw input data and its outcomes. First of all, divide the data into a training dataset, and a test dataset, then the network is trained using the training data set while the test dataset is used to predict outcomes or to see the accuracy of our model.

In supervised learning, consistency is what we accomplish, as model perfection is typically high. Since we already have desirable output in our dataset, the training time taken is less. In some cases, supervised learning models, to obtain the highest possible accuracy, reverse the performance resultto learn more. The algorithm learns the input patterns that produce the expected output, and it can be used to predictthe correct output of an unknown input once the algorithm is qualified.

b. Unsupervised learning:

The training data set is neither categorized nor labeled inthe dataset of unsupervised learning. From an unlabelled data set, Unsupervised learning infers a feature to define a hidden structure. Seeking patterns in the data is the key challenge of unsupervised learning. Once the model can create patterns, then it can predict the pattern of any data set.

c. Reinforcement Learning:

It does not have labeled datasets or data-related outcomes, so a given task is performed by learning from experience. It is rewarded with positive reinforcement for every correct action,while for every incorrect action, it is rewarded with negative reinforcement. From this, it learns about the activities thatare to be performed and not. Reinforcement learning is widely used in industrial automation as well as the gaming sector.

4.3 Deep Learning

Deep Learning is a type of computer program that imitates the neuronal network in the brain. It is a subclass of machine learning that combines representation learning and artificial neural networks. Because deep neural networks are used, it is known as deep

learning. Supervised, semi-supervised, or unsupervised learning is possible. Connected layers are used in the construction of deep learning systems.

- First layer: Input Layer
- Last layer: Output Layer
- All layers in between are called Hidden Layers.

There are neurons in every Hidden layer that are interconnected. The input signal that the neuron receives will be processed by it before being transmitted to the layer above it. The weight, bias, and activation function all affect the signal strength that is sent to the neuron in the following layer. The network processes vast amounts of incoming data through many layers, where it can learn ever more complicated characteristics of the data[14].

4.3.1 Convolutional Neural Network (CNN)

The output of a multi-layered neural network called a CNN is determined by extracting progressively complicated features from the data at each layer. CNN is excellent at jobs requiring perception[13].

When practitioners need to extract information from an unstructured data set (such as photos), CNN is frequently used[12].

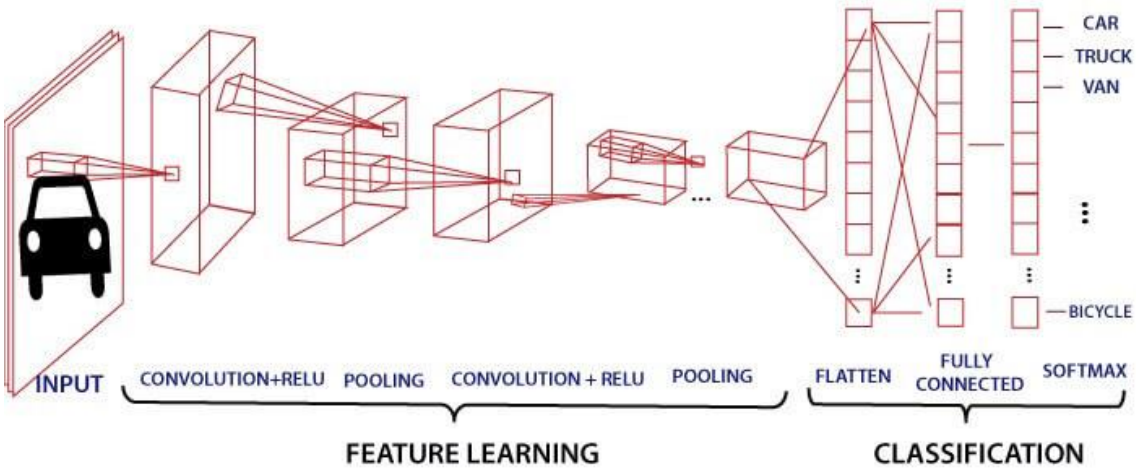


Figure 4.1: Convolutional Neural Network

For instance, if the task is to predict an image :

• The CNN receives an image, say of a car, which is a collection of pixels. Typically, a greyscale image has one layer, whereas a color image has three layers.

• The network will recognize distinctive features, such as the cat's tail, the ear, etc., in the learning (i.e., hidden layers).

• CNN may offer a probability for each image it recognizes after properly learning how to do so. The network's forecast will be the label with the highest probability.

CHAPTER 5

METHODOLOGY

5.1 General

The overall computational pipeline for the proposed work is illustrated in figure 7. It consists of different stages like image acquisition, classification, localization, segmentation, crack width analysis, etc. Each of the stages is explained in the following sections.

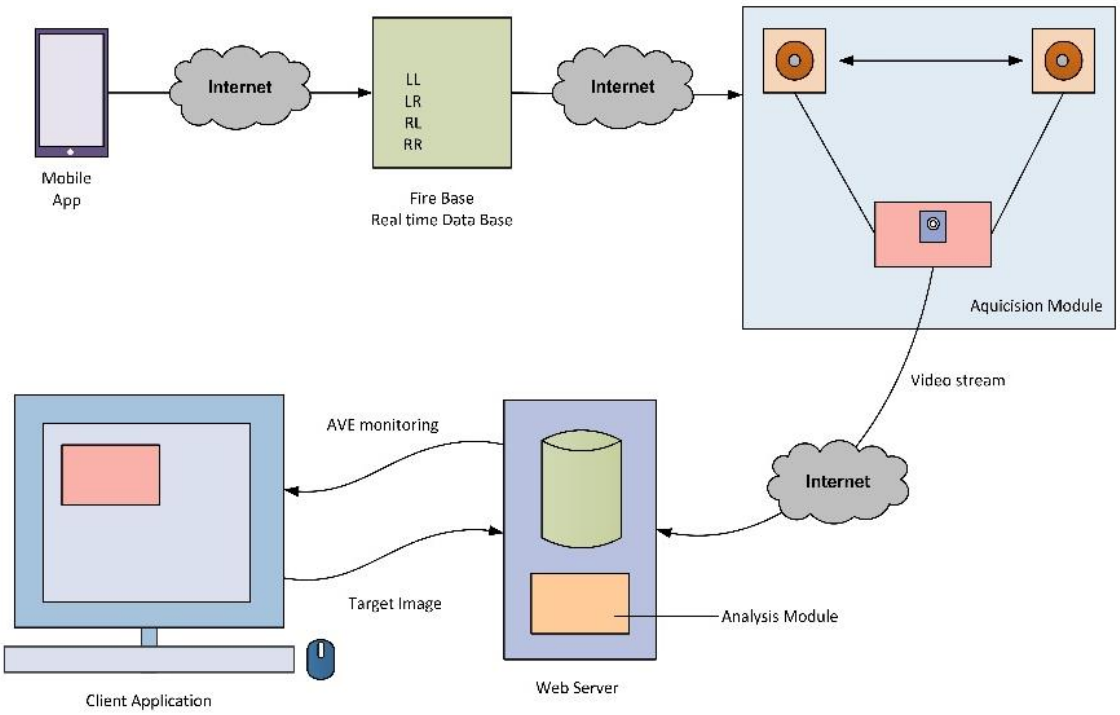


Figure 5.1: Architecture of the proposed method

5.2 Portable Image Acquisition Module

The process of extracting an unprocessed image from an item or scene using an optical device, typically hardware such as cameras, sensors, etc., and converting it into a form that can be processed and analyzed. Since the system cannot perform any meaningful processing without an image, it is both the first and most crucial stage in the workflow sequence. Typically, the technology captures an entirely unprocessed image.

In the proposed system the acquisition module is designed as a portable IoT device. The device can be hung parallel to the member to be observed and using a remote control one can control it to move vertically and horizontally.

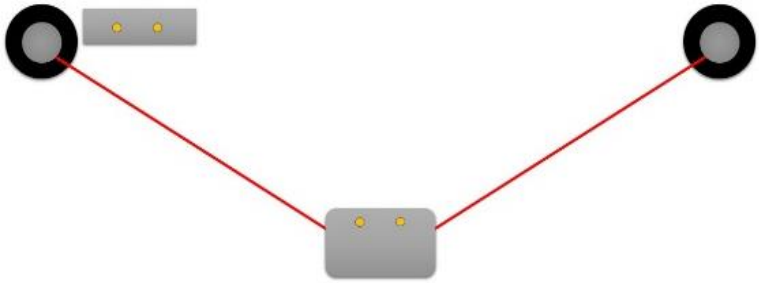


Figure 5.2: Proposed Image Acquisition Module

This movement ensures the complete coverage of the member for examination. The images captured using the devices are then sent to an analysis server over the internet.

5.2.1 Components of Portable Acquisition Module

It comprises two 90rpm dc motors, one is on the left end along with the Motor controller, IoT hardware ESP 8266, and charging dock, and the other one is on the right end. The Wi-Fi camera module and battery are mounted in one hanging unit.

A. ESP Wi-Fi Camera module

Figure 5.3: ESP Wi-Fi Camera module

A compact, inexpensive development board built on the ESP32 platform is called ESP32-CAM. It is the best option for IoT applications, building prototypes, and do-it-yourself projects.

It includes two powerful 32-bit LX6 CPUs, WiFi, classic Bluetooth, and low-power BLE. The range of its primary frequency modification is 80MHz to 240MHz. It is widely applicable in numerous IoT applications. It is appropriate for IoT applications such as

wireless positioning system signals, wireless control, wireless monitoring, and smart home gadgets.

B. ESP 8266

Figure 5.4: ESP 8266

The ESP8266 WiFi Module, a self-contained SOC with an integrated TCP/IP protocol stack, enables any microcontroller to connect to your WiFi network. The ESP8266 is capable of hosting an application or delegating all Wi-Fi networking duties to another application processor. An AT command set firmware is pre-programmed into each ESP8266 module, so you can connect it to your Arduino device and gain WiFi functionality comparable to that of a WiFi Shield. The ESP8266 module is a very affordable board with a sizable and expanding community.

C. L298 2A motor driver

The high-power L298N 2A Based Motor Driver is ideal for driving Stepper Motors and DC Motors. It makes use of the well-known L298 motor driver IC and includes a built-in 5V regulator that it can utilize to power an external circuit. It can direct and speed-control up to 4 or 2 DC motors.

For robotics projects, this motor driver is ideal for controlling motors with microcontrollers, switches, relays, etc. Perfect for powering DC and Stepper motors used in robot arms, line-following robots, micro mice, and other applications.

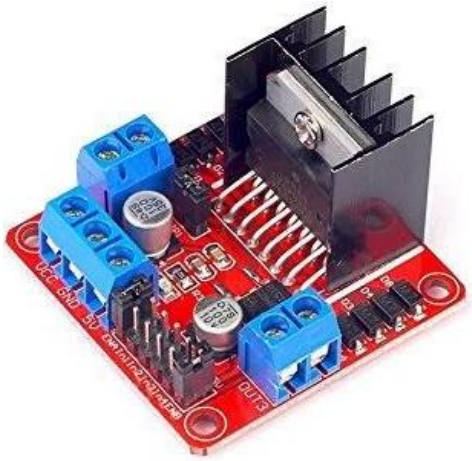


Figure 5.5: L298 2A motor driver

D. 12v 90rpm dc motor

Figure 5.6: 12v 90rpm dc motor

A gear assembly is joined to a DC motor to create a geared motor. Motor speed is measured in terms of shaft revolutions per minute and is referred to as RPM. The gear assembly aids in lowering speed and raising torque.

E. 12v dc supply

5.3 Client Application Module

Generally, a client application serves as an intermediate between the end-user and the system. The proposed system consists of two client applications, one is an Android mobile application and the other one is a web application developed using the flask-python module. The details of these two applications are described below.

5.3.1 Mobile Application

The main purpose of the mobile application is to control the movement of the acquisition module. Currently, the application is developed on the android platform, but in the future, if needed it can be developed on other native platforms like IOS and windows or we can

build it as a cross-platform app. The app facilitates the user to move the camera module in four directions say LEFT, RIGHT, UP, and DOWN, also there is a provision to calibrate the duration of each movement step by setting the required time using an input text field. The input from the user is received by the application and it performs certain actions corresponding to each input. The communication between this controller app and the acquisition module is governed by a real-time database called firebase. On each action performed by the user, the mobile application pushes some value to the firebase and this is fetched by the controller module. Upon evaluating the combination of values pushed by the application the controller module decides where to move. So users can easily control the acquisition module with the help of this application.

5.3.2 Web Application

The web application is one core module that facilitates the analysis of the captured image. Several options are incorporated in the web application, such as live streaming of the video from the acquisition module, taking still images from the stream, controlling the acquisition module and analyzing the crack. Apart from the live stream capturing, users can upload the images already taken. The web application is built using the technologies like HTML, CSS, and JavaScript. The user can input sever parameters for the analysis process through the web application. The structural element and the position of the crack in the member is given as the input parameters. At each event in the web application, an asynchronous HTTP request will be sent to the server. On receiving this request, the server will perform the background computations and return the results as an HTTP response. As result of this background computation,type of the crack as well as the width of the crack is get it as output. The segmented crack image is displayed in the web page. The data format used for the HTTP communication is JSON. The following figures show the screenshot of the web application.

5.4 Server Module

The Server Module governs the computational process in the proposed system. The communications happening at the mobile and web applications is handled by the server module.It receives and responds to the HTTP request from the clients after adequately handling it. The core computations, including image processing and deep learning tasks, are managed on the server. The server module was built using python, mainly using the Flask package. Flask is quite a lightweight server, at the same time, powerful. Using flask

every request received at the server is appropriately handled, and any errors or exceptions occurring while performing the computation are managed in an assertive manner.

5.5 Analysis Module

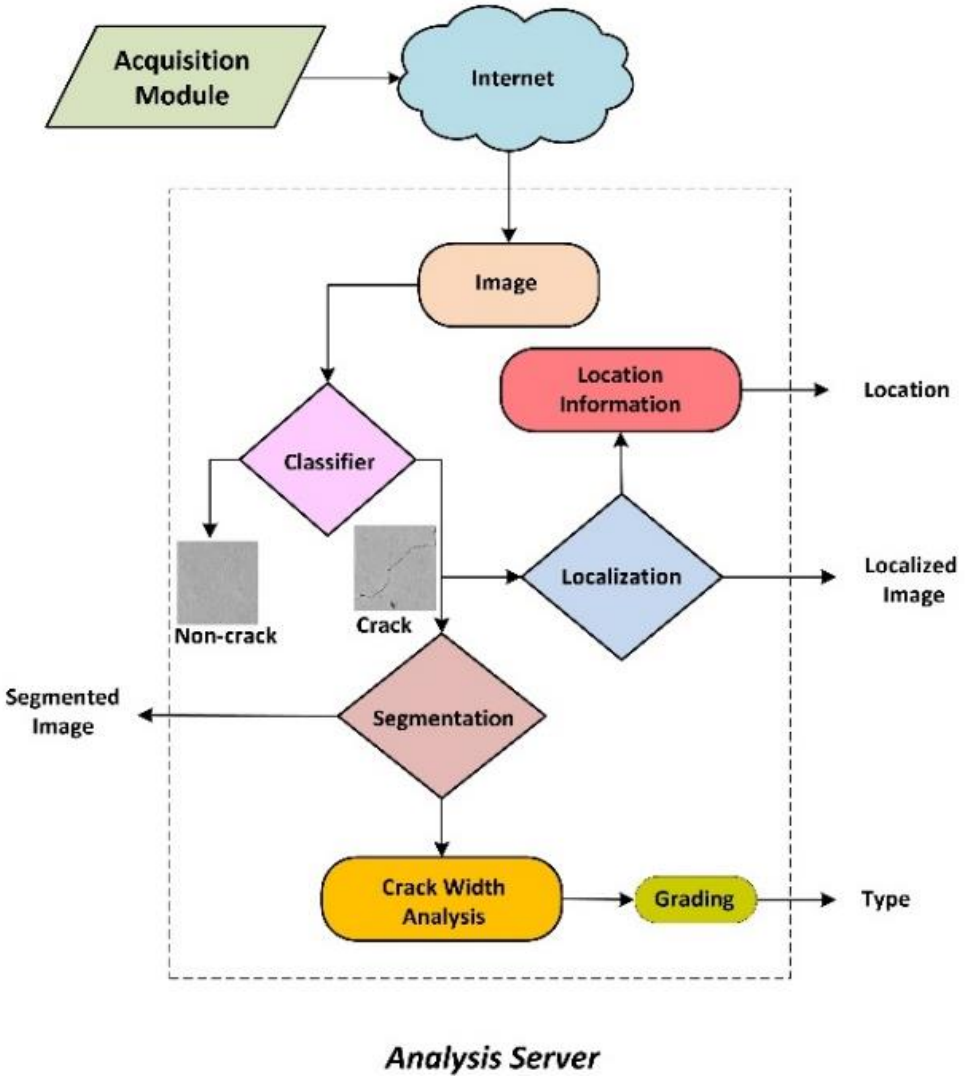


Figure 5.7: Web Application – Live view web page

5.5.1 Classification

It is the process of classifying a given collection of data into different categories, classification can be applied to both organized and unstructured data. Predicting the class of the provided data points is the first step in the procedure. The terms target, label, and classes are frequently used to describe the classes.

Figure 5.8: Data set - Crack and Non-Crack images

The goal of classification predictive modeling is to approximate the mapping function between discrete output variables and input variables. The basic objective is to determine what class or group the new data will belong to.

5.5.2 Localization

A bounding box is drawn around the extent of one or more objects in an image once their location has been determined through localization. A localized image with its position information is obtained as the output after this phase.

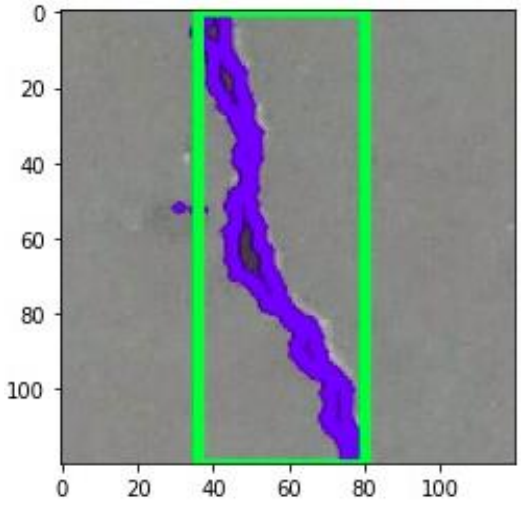


Figure 5.9: Localised Crack image

5.5.3 Segmentation

The method of clustering portions of an image that correspond to the same object class is known as image segmentation. This method is also known as "pixel-level categorization. Label masks are produced as an output to separate pixels with and without cracks. By creating the segmented crack image, it finds the cracks and determines their shape. An output segmented image is obtained at this stage. At this phase, a segmented image is get as output.

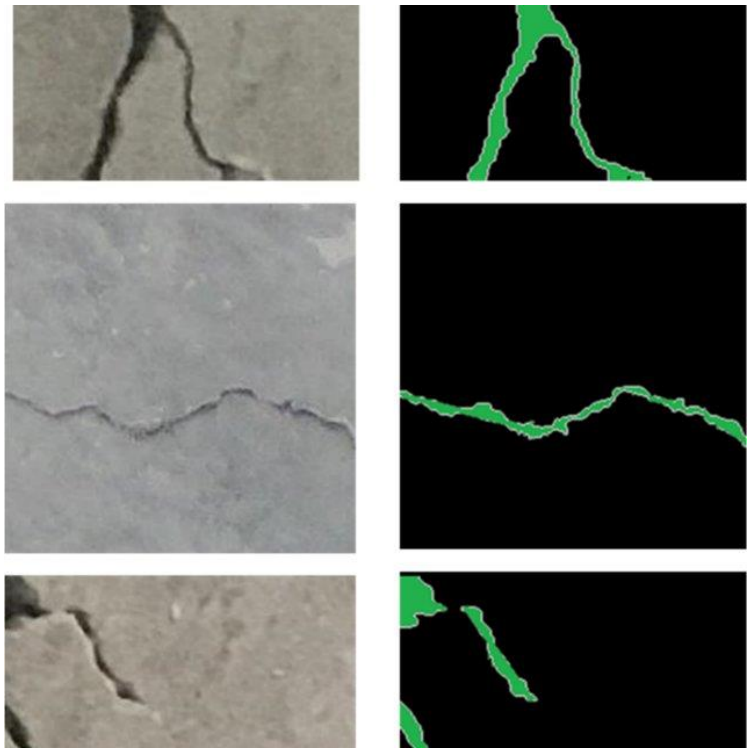


Figure 5.10: Segmented Crack image

5.5.4 Crack Width Analysis

After localization and segmentation next, step is crack width analysis. The width of the crack is determined from the segmented images and is graded based on the width.

5.6 Dataset

The data set comprises images of concrete surface cracks collected from various sources such as Kaggle (dataset online site), self-collected, and datasets available in various literature, etc.

The images so collected are divided into two parts, one for the classification and the other for the segmentation process.

5.6.1 Classification Dataset

The classification data set comprises concrete surface crack images of 40000 images- 20000 non-crack and 20000 crack images.

5.6.2 Segmentation Dataset

The segmentation dataset comprises 1074 nos concrete surface crack images. Total images are divided into training (300 nos) and testing datasets (237nos).

Table 2: Segmentation dataset

Folder	Description
`train_img`	RGB images for training
`train_lab`	binary annotation for training images
`test_img`	RGB images for testing
`test_lab`	binary annotation for testing images

5.7 Tools

5.7.1 Python

Python is an interpreted, general-purpose, dynamic, and high-level programming language. It supports the approach of designing programmes known as object-oriented programming. It offers a large number of high-level data structures and is straightforward and simple to learn. Python is the best language for scripting and quick application development because of its syntax, dynamic typing, and nature as an interpreted language Python supports a variety of programming patterns, including imperative, functional, and object-oriented programming patterns. Python is not designed to be used for a certain task, like web programming. Because it can be utilized online, in the enterprise, in 3D CAD, etc., it is known as a multipurpose programming language.

5.7.2 Flask

Flask is a compact and lightweight Python web framework that facilitates the development of online applications. Since you can quickly create a web application using only one Python file, it allows developers flexibility and is a more approachable framework for beginning developers. Furthermore, Flask is expandable and doesn't demand a specific directory structure or lengthy boilerplate code to start using it. Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have a built-in abstraction layer for database handling, nor does it have form validation support. Instead, Flask supports the extensions to add such functionality to the application.

5.7.3 Open CV

A computer vision and machine learning software library called OpenCV is available for free use. A standard infrastructure for computer vision applications was created with OpenCV in order to speed up the incorporation of artificial intelligence into products. OpenCV makes it simple for businesses to consume and alter the code because it is a BSD-licensed product.

More than 2500 optimised algorithms are available in the collection, including a wide range of both traditional and state-of-art computer vision and machine learning algorithms.

These algorithms can be used to find similar images from an image database, remove red eyes from flash-taken photos, follow eye movements, recognise scenery, and establish markers to overlay. They can also be used to detect and recognise faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce high-resolution images of entire scenes, extract 3D models of objects from stereo cameras, and extract 3D models of objects.

5.7.4 Tensor Flow

Google created the open-source library TensorFlow specifically for deep learning applications. Traditional machine learning is also supported.

Tensors, which are multi-dimensional arrays with more dimensions, are the only type of data that TensorFlow takes. When handling a lot of data, multi-dimensional arrays come in quite helpful.

Data flow graphs with nodes and edges serve as the foundation for TensorFlow's operation. It is considerably simpler to distribute the execution of TensorFlow code using GPUs across a cluster of computers because the execution mechanism takes the form of graphs

5.7.5 Android Studio

Android studio is generally used to create apps for Android phones, tablets, Android Wear, Android TV, and Android Auto, which offers a single development environment. The entire project can be broken into functional pieces that you can separately create, test, and debug using structured code modules.

5.8 Validation

Validation is a process of verifying and documenting with a high degree of assurance that specific equipment will perform consistently according to the requirements. Elcometer is the equipment that I used for the validation purpose. The Elcometer 900 is a fairly straightforward gauge used to gauge the width of concrete cracks.By counting the number of graduations on the illuminated graded scale inside the 50x magnification microscope, the user may instantly determine the crack width.The microscope has graduations in both metric and imperial systems.

The Elcometer and the proposed concrete crack analysis system were both installed in the concrete lab at TKM to analyse the crack. Therefore, the results, ie, crackwidth shown by the two equipents, were compared in order to validate the system.

Figure 5.11: Elcometer

CHAPTER 6

RESULT

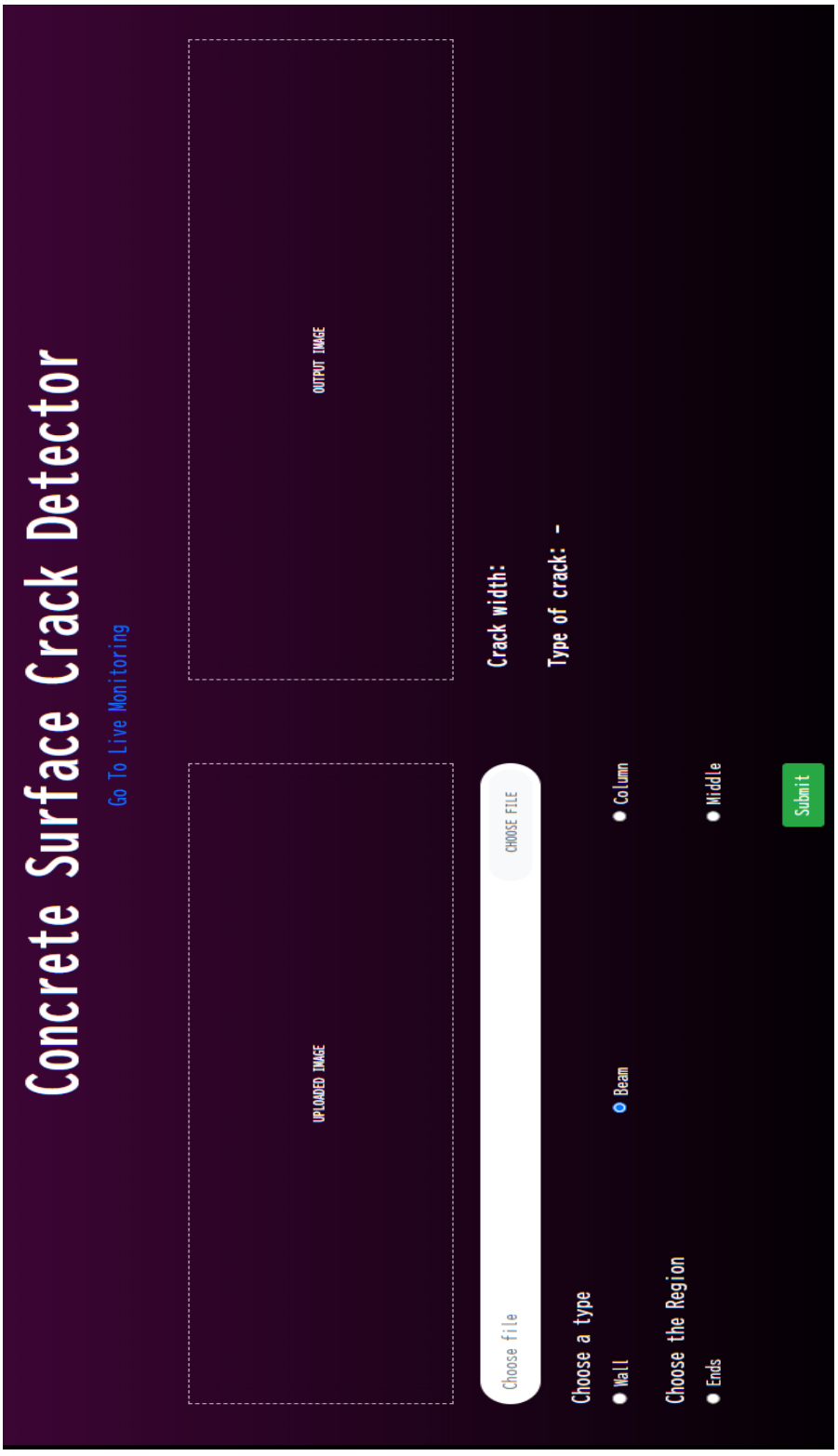


Figure 6.1: Web Application – Output web page

Figure 6.1 and 6.2 depicts the web interface for output and the monitoring of concrete surface in real time.

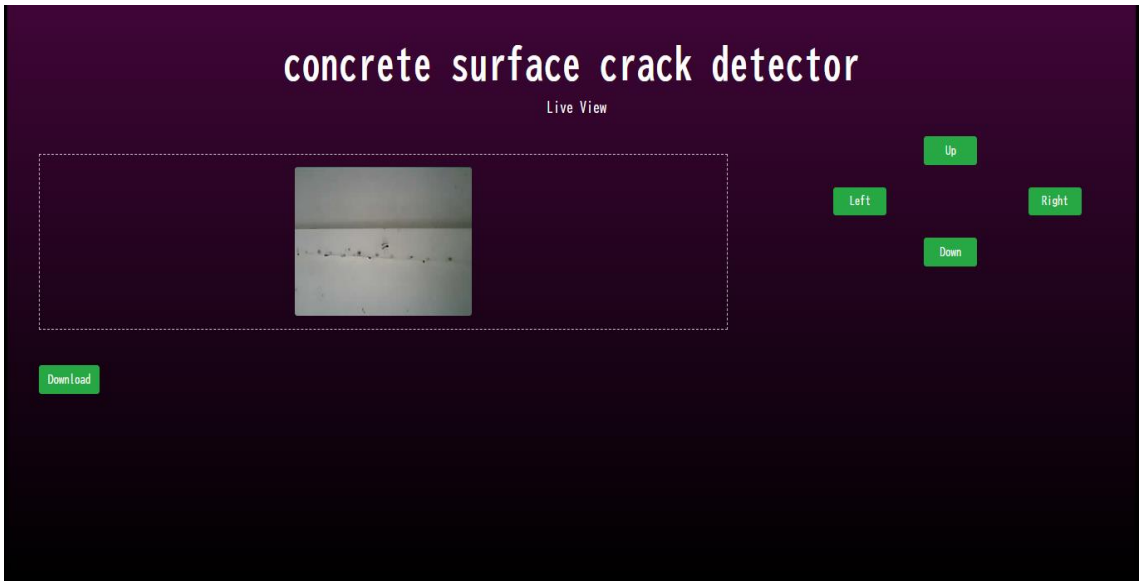


Figure 6.2 : Web Application – Live view web page

Figure 6.3 shows the android mobile application developed for the interaction to analysis.

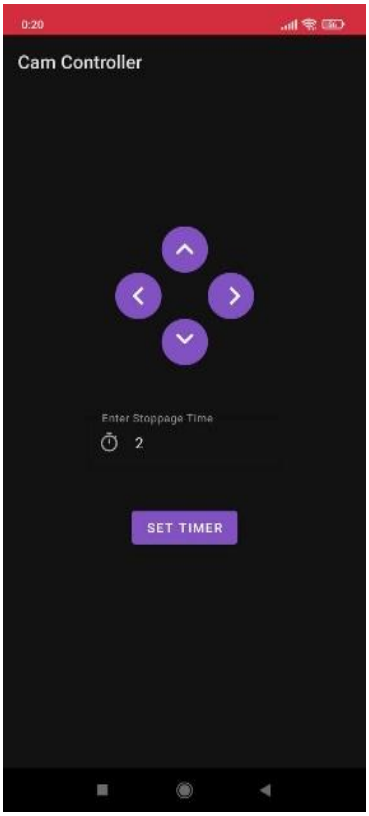


Figure 6.3: Andriod Mobile Application – Cam controller

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm. Confusion matrices are useful because they give direct comparisons of values like True Positives, False Positives, True Negatives and False Negatives. Fig 6.4 shows the confusion matrix of the classification algoritham that we are developed. The accuracy of the CNN model is 97.75%.

$$Accuracy = \frac{196 + 195}{400} = 97.75\%$$

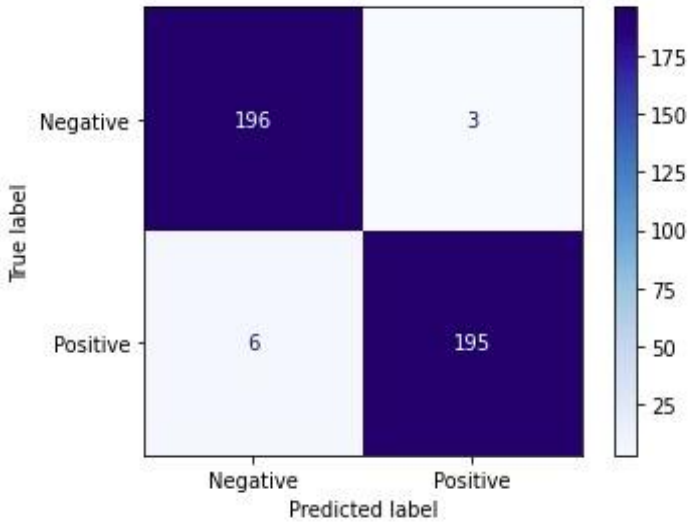


Figure 6.4: Confusion Matrix

The system is validated with real time concrete surface cracks by comparing the values that are obtained by measuring using Elcometer as shown below in figure 6.5 and 6.6.

Figure 6.5: Crack width measurning using Elcometer

Figure 6.6: Crack Analysis using proposed equipment

CHAPTER 7
CONCLUSION

In this project my proposal is to **Design And Development Of Concrete Surface Crack Analysis System**. Cracks are the representation of the total extent of the damage, or they may point to problems of greater magnitude. Crack detection is an important task in the inspection of concrete structures to ensure their protection and serviceability. As the arising concern regarding these cracks in structures, is an inevitable problem in the field of structural engineering.The proposal spans the development of the end-to-end system for concrete surface crck analysis. The major components includes an aquisiation module,an analysis module and client application module. The acuisation modules is designed and developed as an IoT device that can be controlled over internet using a mobile application, and it is intended to move over the suface and capture the cracks. The analysis module is integrated with the webserver, build using python programming and flask library. The core analysis process was implmented using computer vision and deeplearning algorithm.The classification of crack images to cracked or non-crack was done by convolutional neural networks, the segmentation and localizaion is carried out with image processing. The client applications have two versions,one is a web app and another a mobile app which is used to control the acusiation and and to interect to analysis. The solution proposed was validated using different realtime situations. The proposed model for crack detection can be easily remodified and modeled for other survilance need.

CHAPTER 8

REFERENCES

1. Ali, L. *et al.* (2021) 'Performance evaluation of deep CNN-based crack detection and localization techniques for concrete structures, *Sensors*, 21(5), pp. 1–22. DOI: 10.3390/s21051688.
2. Ali, R. *et al.* (2022) 'Structural crack detection using deep convolutional neural networks', *Automation in Construction*. Elsevier B.V., 133(September 2021), p. 103989. DOI: 10.1016/j.autcon.2021.103989.
3. Chen, K. *et al.* (2019) 'Improved Crack Detection and Recognition Based on Convolutional Neural Network', *Modelling and Simulation in Engineering*, 2019. DOI: 10.1155/2019/8796743.
4. Choudhary, G. K., and Dey, S. (2012) 'Crack detection in concrete surfaces using image processing, fuzzy logic, and neural networks, *2012 IEEE 5th International Conference on Advanced Computational Intelligence, ICACI 2012*, pp. 404–411. doi: 10.1109/ICACI.2012.6463195.
5. Esmail Shahrokhinasab, Nima Hosseinzadeh, Armin Monirabbasi, and Sadegh Torkaman. Performance of image-based crack detection systems in concrete structures. Journal of Soft Computing in Civil Engineering, 4(1):127–139, 2020.
6. Feng, C. *et al.* (2019) 'Structural Damage Detection using Deep Convolutional Neural Network and Transfer Learning Structural Damage Detection using Deep Convolutional Neural Network and Transfer Learning', (August). doi: 10.1007/s12205-019-0437-z.
7. Darwin, D. (1998) 'Causes, Evaluation, and Repair of Cracks in Concrete Structures',
8. Fujita, Y. and Hamamoto, Y. (2011) 'A robust automatic crack detection method from noisy concrete surfaces', *Machine Vision and Applications*, 22(2), pp. 254. doi: 10.1007/s00138-009-0244-5.
9. Islam, M. M. M., and Kim, J. (2019) 'Vision-Based Autonomous Crack Detection of Concrete Structures Using a Fully Convolutional Encoder-Decoder Network', pp. 1–12.
10. Janani, S. (2018) 'Measurement of Crack Detection in Concrete Image Using Image processing', *Journal of Critical Reviews* 6(2), pp. 631–638.
11. Baohua Shan, Shijie Zheng, and Jinping Ou. A Stereovision-based Crack Width Detection Approach for Concrete Surface Assessment. 00(0000):1–10, 2015.
12. Keqin Chen, Amit Yadav, Asif Khan, Yixin Meng, and Kun Zhu. Improved Crack Detection and Recognition Based on Convolutional Neural Network. Modeling and Simulation in Engineering, 2019.
13. Keiron and Ryan Nash. An introduction to convolutional neural networks,2015.
14. Kim, B. and Cho, S. (2018) 'Automated vision-based detection of cracks on concrete surfaces using a deep learning technique', *Sensors (Switzerland)*, 18(10). DOI: 10.3390/s18103452.
15. Kim, J. J., Kim, A. R. and Lee, S. W. (2020) 'Artificial neural network-based automated crack detection and analysis for the inspection of concrete structures, *Applied Sciences (Switzerland)*, 10(22), pp. 1–13. doi: 10.3390/app10228105.
16. Li, S. and Zhao, X. (2019) 'Image-Based Concrete Crack Detection Using Convolutional

Neural Network and Exhaustive Search Technique', *Advances in Civil Engineering*, 2019(Ml). doi: 10.1155/2019/6520620.
17. Lins, R. G., and Givigi, S. N. (2016) 'Automatic Crack Detection and Measurement Based on Image Analysis, *IEEE Transactions on Instrumentation and Measurement*, 65(3), pp.583–590. doi: 10.1109/TIM.2015.2509278.
18. Mandal, V., Uong, L. and Adu-Gyamfi, Y. (2019) 'Automated Road Crack Detection Using Deep Convolutional Neural Networks, *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*. IEEE, pp. 5212–5215. DOI: 10.1109/BigData.2018.8622327.
19. Mahesh, A. (2018) 'A Survey on Crack Detection Using Image Processing Techniques and Deep Learning Algorithms', *Alexandria Engineering Journal* 118(8), pp. 215–220.
20. Mohan, A. and Poobal, S. (2018) 'Crack detection using image processing: A critical review and analysis', *Alexandria Engineering Journal*, 57(2), pp. 787– 798.doi: 10.1016/j.aej.2017.01.020.
21. Mukkamala S.N.V. Jitendra, P. Naga Srinivasu, A. Shanmuk Srinivas, Alavala Nithya, and Sai Kiran Kandulapati. Crack detection on concrete images using classification techniques in machine learning. Journal of Critical Reviews, 7(9):1236–1241, 2020.
22. Nhat Duc Hoang. Detection of Surface Crack in Building Structures Using Image Processing Technique with an Improved Otsu Method for Image Thresholding. Advances in Civil Engineering, 2018, 2018.
23. Silva, W. R. L. da and Lucena, D. S. de (2018) 'Concrete Cracks Detection Based on Deep Learning Image Classification', *Proceedings*, 2(8), p. 489. doi:10.3390/icem18-05387.
24. Su, T. (2013) 'Application of Computer Vision to Crack Detection of Concrete Structure', *Proceedings* pp. 457–461. doi: 10.7763/IJET.2013.V5.596.
25. Suguru Yokoyama and Takashi Matsumoto. Development of an automatic detector of cracks in concrete using machine learning. Procedia Engineering, 171:1250–1255, 2017. The 3rd International Conference on Sustainable Civil Engineering Structures and Construction Materials - Sustainable Structures for Future Generations.
26. Tomoyuki Yamaguchi and Shuji Hashimoto. Fast crack detection method for large-size concrete surface images using percolation-based image processing. Machine Vision and Applications, 21(5):797–809, August 2010.
27. Yao, G. *et al.* (2018) 'Crack Detection of Concrete Surface Based on Newline Convolutional Neural Networks, *Proceedings - International Conference on Machine Learning and Cybernetics*, 1, pp. 246–250. doi:10.1109/ICMLC.2018.8527035.
28. Yusuke Fujita and Yoshihiko Hamamoto. A robust automatic crack detection method from noisy concrete surfaces. Machine Vision and Applications,22(2):245–254, 2011.
29. Zeaon Backus. Concrete Cracking: Evaluating Width, Depth & Movement.
30. ZAlex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

LIST OF PUBLICATIONS

Vijayan, V., Joy, C. M., & Shailesh, S. S. (2021). A Survey on Surface Crack Detection in Concretes using Traditional, Image Processing, Machine Learning, and Deep Learning Techniques. *ICCISc 2021 - 2021 International Conference on Communication, Control and Information Sciences, Proceedings*. https://doi.org/10.1109/ICCISc52257.2021.9484914

APPENDIX

Import statements

```
import os
import cv2
import numpy as np
import pandas as pd
from PIL import Image
from sklearn.preprocessing import LabelEncoder
import tensorflow as tf
from tensorflow.python.keras import layers
from tensorflow.python.keras import models
import sklearn.metrics as metrics
from sklearn.metrics import ConfusionMatrixDisplay
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
from tensorflow import keras
import random
from sklearn.model_selection import train_test_split
```

Load Image and Label

```
X = []
y = []

base_path='./dataset/'
source_path=base_path
for child in os.listdir(source_path):
    i=0
    sub_path = os.path.join(source_path, child)
    if os.path.isdir(sub_path):
        for data_file in os.listdir(sub_path):
            X_i = Image.open(os.path.join(sub_path, data_file))
            X_i = np.array(X_i.resize((120,120))) / 255.0
            X.append(X_i)
            y.append(child)
```

LabelEncoder

```
encoder = LabelEncoder()
y = encoder.fit_transform(y)
```

Train Test Split

```
X_train, X_test, y_train, y_test = train_test_split(np.array(X), np.array(y),
                                    test_size=0.2, random_state=42)
```

Training

```
random.seed(42)
np.random.seed(42)
#tf.random.set_seed(42)
tf.random.set_random_seed(42)
```

```
cnnModel = models.Sequential()
cnnModel.add(layers.Conv2D(10, (3,3), activation="relu",
                    input_shape=(120,120,3)))
cnnModel.add(layers.MaxPooling2D((2,2)))
cnnModel.add(layers.Conv2D(64, (3,3), activation="relu"))
cnnModel.add(layers.MaxPooling2D((2,2)))
cnnModel.add(layers.Conv2D(64, (3,3), activation="relu"))
cnnModel.add(layers.Flatten())
cnnModel.add(layers.Dense(64, activation="relu"))
cnnModel.add(layers.Dense(32, activation="relu"))
cnnModel.add(layers.Dense(1, activation="sigmoid"))
cnnModel.summary()
cnnModel.compile(optimizer="adam",
loss="binary_crossentropy",metrics=["accuracy"])
cnnModel.fit(X_train, y_train, epochs=20, batch_size=32)
testLoss, testAccuracy = cnnModel.evaluate(X_test, y_test)
cnnModel.save('cnn_model.h5')
```

Prediction & Segmentation

```
X = []
import random
import os
from pathlib import Path
import cv2
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt

from tensorflow import keras
from PIL import Image

def pil_to_cv2(pil_image):
    open_cv_image = np.array(pil_image)
    open_cv_image = open_cv_image[:, :, ::-1].copy()
    return open_cv_image

def cv_to_pil_image_converter(image):
    """
    Converts opencv type image to pil image
    :param image: pil image that is to be converted
    """
    # You may need to convert the color.
    img = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    im_pil = Image.fromarray(img)
    return im_pil
def cv_to_pil_image_converter_g(image):
    """
    Converts opencv type image to pil image
    :param image: pil image that is to be converted
    """
```

```
    # You may need to convert the color.
    img = cv2.cvtColor(image, cv2.COLOR_GRAY2RGB)
    im_pil = Image.fromarray(img)
    return im_pil
def most_common(lst):
    return max(set(lst), key=lst.count)
def run(path):
    X = []
    model = keras.models.load_model('cnn_model.h5')
    X_ = Image.open(path)
    X_ = X_.resize((120, 120))
    X_i = np.array(X_) / 255.0
    X.append(X_i)
    clas = model.predict_classes([X])[0][0]
    img = pil_to_cv2(X_)
    if clas ==1:
        src = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        # Set threshold and maxValue
        thresh = 127
        maxValue = 255
        # Basic threshold example
        th, dst = cv2.threshold(src, thresh, maxValue, cv2.THRESH_BINARY);
        dst = 255 - dst
        kernel = np.ones((3, 4), np.uint8)
        image = cv2.erode(dst, kernel)
        # Find Contour
        countours, hierarchy = cv2.findContours(image, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
        cnt = sorted(countours, key=cv2.contourArea)
        width=0
        old_width=0
        if len(cnt)>0:    # Draw Contour
            cv2.drawContours(img, cnt, -1, (0, 0, 255), 2, lineType=cv2.LINE_AA)
            rect = cv2.boundingRect(cnt[-1])
            lst1 = [item[0][0] for item in cnt[-1]]
            common= lst1.index(most_common(lst1))
            random_x=cnt[-1][common][0][0]
            random_y=cnt[-1][common][0][1]
            width=0
            old_width_x=0
            for i in cnt[-1]:
                if random_x==i[0][0]:
                    width=abs(random_y-i[0][1])
                    #print("y----------",i[0])
                    if width <= old_width_x:
                        continue
                    else:
                        old_width_x=width
            lst2 = [item[0][1] for item in cnt[-1]]
            common= lst2.index(most_common(lst2))
```

```
            random_x=cnt[-1][common][0][0]
            random_y=cnt[-1][common][0][1]
            width=0
            old_width=0
            for i in cnt[-1]:
                if random_y==i[0][1]:
                    width=abs(random_x-i[0][0])
                    #print("y----------",i[0])
                    if width <= old_width:
                        continue
                    else:
                        old_width=width
            if old_width>old_width_x:
                old_width=old_width_x

            x, y, w, h = rect
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
        else:
            font = cv2.FONT_HERSHEY_SIMPLEX
            # org
            org = (5, 50)
            # fontScale
            fontScale = 0.2
            # Blue color in BGR
            color = (255, 0, 0)
            # Line thickness of 2 px
            thickness = 1
            # Using cv2.putText() method
            img = cv2.putText(img, 'Undetectable Complex Crack', org, font,
                              fontScale, color, thickness, cv2.LINE_AA)
        # plt.imshow(img)
        # plt.show()
        img=cv_to_pil_image_converter(cv2.resize(img,(120,120)))
        image=cv_to_pil_image_converter_g(cv2.resize(image,(120,120)))
        Dummy = Image.new("RGB", (240, 120), "white")
        Dummy.paste(img, (0, 0))
        Dummy.paste(image, (120, 0))
        return Dummy,old_width
    else:
        font = cv2.FONT_HERSHEY_SIMPLEX

        # org
        org = (10, 50)
        # fontScale
        fontScale = 0.4
        # Blue color in BGR
        color = (255, 0, 0)
        # Line thickness of 2 px
        thickness = 1
        # Using cv2.putText() method
```

```
        img = cv2.putText(img, 'No Crack Found', org, font,
                          fontScale, color, thickness, cv2.LINE_AA)
        # plt.imshow(img)
        # plt.show()
        return img,0
```

Flask server

```
from flask import Flask, request,
jsonify,Response,send_file,render_template,redirect,url_for
import cv2
import numpy as np
from inference import run
import uuid
app = Flask(__name__)
@app.route('/')
def index():
    return render_template("index.html")
@app.route('/process')
def process():
    return render_template("index.html",init='no')
@app.route('/view')
def view():
    return render_template("view.html")
@app.route('/detect/', methods=['POST'])
def detect():
    print('entry point')
    myuuid = uuid.uuid4()
    try:
        if request.files['upload'].filename != '':
            f=  request.files['upload']
            f.save("./images/input.jpg")
    except:
        pass
    if True:
        img_path = "./images/input.jpg"
        print('read image')
        res,width = run(img_path)
        print('prediction')
        res_path = "D:/WebApp/latest/pythonapp/static/output/res_"+ str(myuuid)+".jpg"
        res.save(res_path)
        data={'width':str(width),"file_name":"res_"+ str(myuuid)+".jpg"}
        return jsonify(data)
@app.route('/image/', methods=['POST'])
def image():
    print("entry okay")
    cap = cv2.VideoCapture('http://192.168.0.102:81/stream')
    print("video open")
    ret, frame = cap.read()
    print("frame readed")
    cv2.imwrite("./images/input.jpg",frame)
```

```
    cv2.imwrite("./static/input.jpg",frame)
    cap.release()
    cv2.destroyAllWindows()
    print("image saved")
    return redirect(url_for('process'))
if __name__ == '__main__':
    app.run(host='localhost', port=8060)
```

Home Page (web)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>cscd</title>
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link
      href="https://fonts.googleapis.com/css2?family=BIZ+UDGothic:wght@400;700&fa
mily=Lobster&display=swap"
      rel="stylesheet"
    />
    <link
      rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css"
      integrity="sha384-
zCbKRCUGaJDkqS1kPbPd7TveP5iyJE0EjAuZQTgFLD2ylzuqKfdKlfG/eSrtxUkn"
      crossorigin="anonymous"
    />
    <link rel="stylesheet" href="{{url_for('static', filename='styles/index.css')}}" />
  </head>
  <body>
    <div class="container-fluid py-5">
      <!-- For demo purpose -->
      <header class="text-white text-center">
        <h1 class="display-4 font-weight-bold capitalize">
          Concrete Surface Crack Detector
        </h1>
        <p class="lead mb-0"><a href='view'>Go To Live Monitoring</a></p>
      </header>
      <div class="row py-4">
        <div class="col-lg-6 px-5">
          <form enctype="multipart/form-data" id="fileUploadForm">
          <!-- Uploaded image area-->
          <div class="image-area image-area-input my-4">
            <img
              id="imageResult"
                              {% if init %}
                                    src="{{url_for('static',filename='input.jpg')}}"
```

```
                    {% else %}
                        src="#"
                    {% endif %}
    alt=""
    class="img-fluid rounded shadow-sm mx-auto d-block image-area-img"
   />
</div>
<!-- Upload image input-->
<div class="input-group px-2 py-2 rounded-pill bg-white shadow-sm">
  <input
    id="upload"
    type="file"
    name="upload"
    onchange="readURL(this);"
    class="form-control border-0"
  />
  <label
    id="upload-label"
    for="upload"
    class="font-weight-light text-muted"
    >Choose file</label
  >
  <div class="input-group-append">
    <label for="upload" class="btn btn-light m-0 rounded-pill px-4">
      <i class="fa fa-cloud-upload mr-2 text-muted"></i
      ><small class="text-uppercase font-weight-bold text-muted"
        >Choose file</small
      ></label
    >
  </div>
</div>

            <h5 class="type__title">Choose a type</h5>
<div class="type__select">
  <div class="form-check">
    <input
      class="form-check-input"
      type="radio"
      name="type"
      id="wall"
      value="wall"
      onclick="update_subtype()"
      checked
    />
    <label class="form-check-label type__label" for="wall">
      Wall
    </label>
  </div>
  <div class="form-check">
    <input
```

```
      class="form-check-input"
      type="radio"
      name="type"
      id="beam"
      value="beam"
      onclick="update_subtype()"
     />
     <label class="form-check-label type__label" for="beam">
      Beam
     </label>
    </div>
    <div class="form-check">
     <input
      class="form-check-input"
      type="radio"
      name="type"
      id="column"
      value="column"
      onclick="update_subtype()"
     />
     <label class="form-check-label type__label" for="column">
      Column
     </label>
    </div>
   </div>
   <h5 class="type__title">Choose the Region</h5>
   <div class="type__select">
    <div class="form-check" id="div_end">
     <input
      class="form-check-input"
      type="radio"
      name="sub_type"
      id="end"
      value="end"
      onclick="update_crack_type()"
     <label class="form-check-label type__label" for="end">
      Ends
     </label>
    </div>
    <div class="form-check" id="div_middle">
     <input
      class="form-check-input"
      type="radio"
      name="sub_type"
      id="middle"
      value="middle"
      onclick="update_crack_type()
     <label class="form-check-label type__label" for="middle">
      Middle
     </label>
```

```
    </div>
    <div class="form-check" id="div_center">
      <input
        class="form-check-input"
        type="radio"
        name="sub_type"
        id="center"
        value="center"
        onclick="update_crack_type()"
      />
      <label class="form-check-label type__label" for="center">
        Center
      </label>
    </div>
    <div class="form-check" id="div_corner">
      <input
        class="form-check-input"
        type="radio"
        name="sub_type"
        id="corner"
        value="corner"
        onclick="update_crack_type()"
      />
      <label class="form-check-label type__label" for="corner">
        Corner
      </label>
    </div>
  </div>
  <div class="d-flex mt-5">
    <button type="button" id="submit" class="btn btn-success ml-auto">
      Submit
    </button>
  </div>
</form>
</div>
<div class="col-lg-6 px-5">
  <!-- Output image area-->
  <div class="image-area image-area-output my-4">
    <img
      id="imageOutputResult"
      src="#"
      alt=""
      class="img-fluid rounded shadow-sm mx-auto d-block"
    />
  </div>
              <div class="type__crack">
    <label for="typeCrack">Crack width:<p id="crack_width"></p></label>
  </div>
  <div class="type__crack">
    <label for="typeCrack">Type of crack:</label>
```

```
          <p id="crack_result">diagonal</p>
        </div>
       </div>
      </div>
    </div>
    <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
  <script src="{{url_for('static', filename='styles/index.js')}}"></script>
  <script>
    $('#submit').click(function(){
      var form = $('#fileUploadForm')[0];
    var data = new FormData(form);
    $.ajax({
        url: "http://localhost:8060/detect/",
        type: 'POST',
        data: data,
        enctype: 'multipart/form-data',
        cache: false,
        contentType: false,
        processData: false,
        success: function(res) {
          if(res['width'] != "0"){
            update_crack_submit();
          }else{
            $('#crack_result').html('Crack Not Detected')
          var file_name = res['file_name'];
$('#crack_width').html(res['width'] + " mm");
           console.log("sucess");
          var outputImg = document.getElementById('imageOutputResult');
          outputImg.src = 'http://localhost:8060/static/output/'+file_name;
        },
        error :function(err)
        {
          console.write(err);
        }
      });
    });
  </script>
<script>
function update_subtype(){
  $('#crack_result').html('-');
  if($('input:radio[name=type]:checked').val() == "wall"){
    $('#div_corner').show();
    $('#div_end').hide();
    $('#div_middle').hide();
    $('#div_center').show();
  }else if($('input:radio[name=type]:checked').val() == "beam"){
    $('#div_corner').hide();
    $('#div_end').show();
    $('#div_middle').show();
    $('#div_center').hide();
```

```
  }else if($('input:radio[name=type]:checked').val() == "column"){
    $('#div_corner').hide();
    $('#div_end').show();
    $('#div_middle').show();
    $('#div_center').hide();
  }
}
update_subtype();
</script>
<script>
function  update_crack_submit(){
  if($('#wall').prop('checked') ==true  ){
    if($('#corner').prop('checked') == true){
      $('#crack_result').html('Compression Crack');
    }
    if($('#center').prop('checked') == true){
      $('#crack_result').html('Shear Crack');
    }
  }
  else if($('#beam').prop('checked') ==true  ){
    if($('#end').prop('checked') == true){
      $('#crack_result').html('Shear Crack');
    }
    if($('#middle').prop('checked') == true){
      $('#crack_result').html('Flexural Crack');
    }
  }
  else if($('#column').prop('checked') ==true  ){
    if($('#end').prop('checked') == true){
      $('#crack_result').html('column - end');
    }
    if($('#middle').prop('checked') == true){
      $('#crack_result').html('column - middle');
    }
  }else{
    $('#crack_result').html(' - ');
  }
}
</script>
  </body>
</html>
```

Live View Page

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>cscd - View</title>
```

```
  <link rel="preconnect" href="https://fonts.googleapis.com" />
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
  <link
href="https://fonts.googleapis.com/css2?family=BIZ+UDGothic:wght@400;700&family=Lobster&display=swap"
    rel="stylesheet" />
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css"
    integrity="sha384-
zCbKRCUGaJDkqS1kPbPd7TveP5iyJE0EjAuZQTgFLD2ylzuqKfdKlfG/eSrtxUkn"
crossorigin="anonymous" />
  <link rel="stylesheet" href="{{url_for('static', filename='styles/index.css')}}" />
  <style>
    .dir {
      min-width: 80px;
    }
  </style>
</head>
<body>
  <div class="container-fluid py-5">
    <!-- For demo purpose -->
    <header class="text-white text-center">
      <h1 class="display-4 font-weight-bold capitalize">
        concrete surface crack detector
      </h1>
      <p class="lead mb-0">Live View</p>
    </header>
    <div class="row py-4">
      <div class="col-lg-8 px-5">
        <div class="image-area image-area-input my-4">
          <img id="live_image" src="http://192.168.0.102:81/stream" alt=""
            class="img-fluid rounded shadow-sm mx-auto d-block image-area-img" >
        </div>
        <div class="d-flex mt-5">
          <form onsubmit="video_off()" method ="post" action="/image/">
          <button type="submit" class="btn btn-success ml-auto dir" ">
            Download
          </button>
            </form>
        </div>
      </div>
      <div class="col-lg-4 px-5">
        <div class="row">
          <div class="col-lg-4 px-5"></div>
          <div class="col-lg-4 px-5">
            <button type="button" id="up" onclick="update_direction('up')" class="btn btn-
success ml-auto dir">
              Up
            </button>
          </div>
```

```
        <div class="col-lg-4 px-5"></div>
       </div>
       <div class="row" style="padding: 30px">
        <div class="col-lg-4 px-5">
         <button type="button" id="left" onclick="update_direction('left')" class="btn
btn-success ml-auto dir">
           Left
         </button>
        </div>
        <div class="col-lg-4 px-5"></div>
        <div class="col-lg-4 px-5">
         <button type="button" id="right" onclick="update_direction('right')" class="btn
btn-success ml-auto dir">
           Right
         </button>
        </div>
       </div>
       <div class="row">
        <div class="col-lg-4 px-5"></div>
        <div class="col-lg-4 px-5">
         <button type="button" id="Down" onclick="update_direction('down')"
class="btn btn-success ml-auto dir">
           Down
         </button>
        </div>
        <div class="col-lg-4 px-5"></div>
       </div>
     </div>
    </div>
    <div class="row py-4">
     <div class="col-lg-6 px-5">
     </div>
    </div>
  </div>
  <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
   <script src="https://www.gstatic.com/firebasejs/8.3.1/firebase-app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/8.3.1/firebase-database.js"></script>
   <script src="{{url_for('static', filename='styles/file_save.js')}}"></script>
  <script>
   const firebaseConfig = {
             apiKey: "AIzaSyDV0H9NzChPSTOqfDkGdEiP2yFm-AWADfc",
             authDomain: "cammotor-46952.firebaseapp.com",
             databaseURL: "https://cammotor-46952-default-rtdb.firebaseio.com",
             projectId: "cammotor-46952",
             storageBucket: "cammotor-46952.appspot.com",
             messagingSenderId: "224541203746",
             appId: "1:224541203746:web:082a654591de763f79d97b",
             measurementId: "G-NWLFM23JBQ"
           };
           firebase.initializeApp(firebaseConfig);
```

```
        var database = firebase.database()
  let img = document.querySelector('img');
  $('#download').click(function () {
    let imagePath = img.getAttribute('src');
    let fileName = 'download.png';
    saveAs(imagePath, fileName);
  });
        function video_off()
        {
                $('#live_image').attr('src','');
        }
  function update_direction(dir) {
    if (dir == "up") {
         var one='1';
         var zero='0';
        for(let i=0; i<10; i++) {
                        database.ref('HOME/' + "Device1").set({
                        LL :one,
                        LR :zero,
                        RL :zero,
                        RR :one
                        })
    } else if (dir == "down") {
                database.ref('HOME/' + "Device1").set({
      LL : 0,
      LR : 1,
      RL : 1,
      RR : 0,
                })
    }else if (dir == "right") {
                database.ref('HOME/' + "Device1").set({
      LL : 0,
      LR : 1,
      RL : 0,
      RR : 1,
                })
    }
    else if (dir == "left") {
      database.ref('HOME/' + "Device1").set({
      LL : 1,
      LR : 0,
      RL : 1,
      RR : 0,
                })
    }
  }
  </script>
</body>
</html>
```