**3. Methodology**

Generally, the feature extraction phase for the traditional model employed image preprocessing techniques such as gaussian smoothening, histogram equalization and blending, connected component extraction and image sharpening. Features using the frequency domain of the Fourier transform of the image was also obtained. Table 1 shows the returned image after each operation in the image preprocessing pipeline as well as the frequency domain of the Fourier transform. Image 1 gives a general overview of the pipeline of the preprocessing steps.

**Table 1** Steps involved

|  |  |
| --- | --- |
| **Step** | **Image** |
| Raw Image | ` |
| Conversion to Grayscale, Smoothening and Image Squaring to |  |
| Image Brightening |  |
| Convolution of Image with mean filter of size to obtain mean image towards image differencing |  |
| Image after CCELD thresholding |  |
| Image after applying Skele-Marker denoising technique |  |

Each image was processed in the sequence demonstrated in the pipeline below and at the end of the preprocessing, a naïve particle sum which involved summation of a 10x10 block on the skeleton image. Finally the image was transformed to the frequency domain and the median of a radius of 20 was captured across all images. The idea was that for soils with large particles, we would get much low frequency components thus a wider range of low frequency components and for soils with smaller particles, the reverse would be the case. This is further explained in section 3.5.

**Fig 3.** Preprocessing pipeline for the images

* 1. **Dataset Collection**

The pictures used for this study were obtained from construction site across different states in Nigeria namely; Lagos, Enugu and PortHarcout. The images were obtained by the collective efforts of the Team using a Samsung Galaxy S20+, an Iphone XR and an Infinix Device. Although some images were taking with a different aspect ratio, a goon number of the was taken with the phone aspect ratio set to 1:1 to have a square image which help retain size after image resizing. A total of 7,294 images were obtained and 70% were used for . We had 13 different soils in terms of particle size as well as color as shown in Table 2 below and our splitting was done in such a was as to have a soil not in the training set but with same class as another soil in the training set in the validation and test set. Table 2 well summarizes this distribution. For example in Table 2, sharp sand was used in Training but Red sand was used for only validation and testing. By this method of validation and testing, we ensure firstly that the parameters in our preprocessing pipeline is optimal to ensure that our model learns essential features to distinguish between different USCS classed even if the soil isn’t in our training data. That is essentially the goal of this project.

**Table 2** Dataset Description

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No.** | **Soil** | **Image** | **Use Case** | **USCS Class** |
| 1 | Stone Dust |  |  |  |
| 2 | inches |  |  |  |
| 3 | inches |  |  |  |
| 4 | inches |  |  |  |
| 5 | 1 inch Gravel |  |  |  |
| 6 | 1.5 inches Gravel |  |  |  |
| 7 | 2 inches |  |  |  |
| 8 | Sharp Sand |  |  |  |
| 9 | White Sand |  |  |  |
| 10 | Red Sand |  |  |  |
| 11 | Brown Gravel Big |  |  |  |
| 12 | Brown Gravel Small |  |  |  |
| 13 | Hard Core |  |  |  |

**3.2 The image preproessing Algorithm**

The CCELD technique primarily takes advantage of the fact that the darkest region of an image with a crack will most likely be the cracked region. It involves utilizing a convolution operation on a grayscale image with a smoothening mean filter of size determined from experimentation discussed in section 4.1. The grayscale image is then subtracted from the result of the convolution to obtain a local deviation image. Using each grayscale images’ pixel distribution, a unique threshold value is determined. Based on this value, thresholding is done and the pixels from the grayscale image whose value in the local deviation image are less than or equal to the unique threshold is set as 255 while those greater than the unique threshold are set as 0. The procedures involve in the proposed CCELD algorithm are discussed explicitly in Sub-sections 3.21 – 3.24. All the computation executed in the CCELD algorithm were implemented with the OpenCV library using the python programming language.

**3.2.1 Smoothening, grayscale conversion and resizing**

Smoothening is a very important step in image preprocessing. By smoothening an image prior to applying techniques such as edge detection or thresholding the amount of high-frequency content, such as noise and edges (i.e., the “detail” of an image) is reduced (Rosebrock, 2021). Grayscaling is also a significant step in IP techniques especially in crack detection. It involves the conversion of 3D color image into a 2D black and white image.

The first step involved the conversion of the RGB image into a grayscale image as depicted in Fig. 7b after which blur smoothening with a kernel was employed. The image was then resized to have a height and width of 400 by 400. This size was chosen as very small sizes would lose much information and very large sizes would make computations more expensive. The blur smoothening operation involved assigning a pixel a value based on the mean values in the neighboring kernel with the pixel in the center of the kernel. Considering any neighborhood, the operation on the center pixel value can be expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Where; represents the th row; represents the *th* column, and is any center pixel surrounded by 8 neighboring pixels.

**3.2.2 Contrast and Brightness Enhancement**

Image contrast and brightness enhancement are fundamental techniques that improve the quality of images. These methods adjust the intensity of pixels in an image to make it more visually appealing and easier to interpret. Contrast enhancement emphasizes the differences between light and dark areas, leading to the enhancement of details. Brightness adjustment ensures that images are neither too dark nor too bright.

Some crack images on finished surfaces could have different color of paints and if a crack exists in both parts of the picture, the threshold determined might not work well in the dark region, hence the brightness of the image is enhanced as shown in Fig. 7c using a simple linear formula expressed in Equ. 2. During this operation, the images were clipped to a sum of 255. Thus, the maximum pixel value obtainable was 255. Intuitively, 20% of 255 was added to all the images to ensure a balanced effect on both bright and dark images.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Where; is the returned brightened image, and is the original gray scale image

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| (a) | (b) | (c) |

**Fig. 7** Effect of grayscaling and brightness enhancement on Images (a) Original Image (b) Grayscale image (c) Brightness and Contrast enhanced Image

**3.2.3 Image Differencing**

This process employed another smoothening operation on the grayscale image using a kernel determined after experimentation which was done in section 4.1 to yield a mean image. Thereafter, a difference between corresponding pixels in the grayscale image and the mean image is obtained. This differencing yields negative results as shown in Fig. 8b for possible crack regions since after finding the average value, the corresponding value in the mean image is significantly larger than the pixel value in the gray image due to the large kernel used. Hence, the result from the subtraction of the mean image from the gray image is negative for all possible crack regions. Generally image differencing is used for edge enhancement typically with small kernels (F. Cheevasuvit et al., 1992) but with larger kernels the focus is no longer on finding edges but on finding dark regions (possible cracks). This optimal smoothening kernel was determined from comparison which involved tracking the average IOU score of 115 CCELD aided with Skele-Marker preprocessed images against their ground truth mask at different kernels and the kernel that gave the best results together with other parameters of CCELD was selected.

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| (a) | (b) |

**Fig. 8** Results of Image differencing (a) Grayscale image with selected region (b) Pixel Value for selected region (red square) after image differencing

**3.2.4 Automated thresholding and determination of crack mask**

The next step involved the determination of a threshold in which pixel values less or equal to that threshold is taken to be a crack pixel and hence is assigned a pixel value of 255 (whitest pixel) and pixel values greater were assigned a pixel value of 0 (darkest pixel) resulting in a crack mask image with pixel values having 0 or 255 (the segmented image). Towards automated threshold selection, a histogram analysis of image pixel distribution was done. In images 1-3 of Fig. 9b, the distribution of pixels values falls on the low side as most pixel values are less than 150 indicating a dark colored image with a lot more darker pixels generally than light ones. In making a comparison, an analysis of specific areas (area within the red box) in Fig. 10 is conducted.

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| (a) | (b) |

**Fig. 9** Distribution of Pixel Values on Images (a) Grayscale image (b) Histogram Showing Grayscale Pixel Distribution

In Image 1 of Fig. 10, pixel values are generally small as pixels outside the crack regions ranges from 60 - 80 and pixels in the crack regions have values in range 25 - 40. However, in Image 2 of Fig. 10, pixel values are considerably larger as values outside the crack region ranges from 145 - 190 and pixels in the crack regions have values in range 140 - 130. Due to this difference in range of values, image differencing will yield larger negative difference in image 1 when compared to image 2. As demonstrated in Section 3.2.3, difference in range of -30 to -40 is obtained in Image 3 of Fig. 8 and a difference in range of -50 to -75 is obtained in Image 4 of Fig. 8. With this, it is evident that a single threshold would not be optimal for all kinds of images. This prompted the analysis for an automated threshold selection.

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| (a) | (b) |

**Fig. 10** (a) Grayscale image with crop area indicator (b) Cropped area of image showing Pixel values

Using percentiles and other statistical means, a formula was obtained. The division variable was found to be the most important in the formula. Thus, with the aid of tracking, the average IOU score of CCELD results augmented with Skele-Marker against the ground truth for varying values of the parameter, the best value for the parameter was obtained as 0.4 to determine the best threshold for a picture. The statistical steps towards thresholding are defined;

**i. Mean Computations**

From the initial smoothened grayscale Image before resizing, three variables containing average of percentiles of pixel values are computed for a single image. represents the first percentile of the image pixel values, represents the one-half percentile. The subscripts denotes the percentile value extracted.

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

The can be said to capture the mean of the crack intensity while the can be said to capture the average of pixels close to the crack and far away from the crack but not so far away. The then gets the average of all 9 percentiles. This imitates using a kernel in obtaining the mean image as 9 percentile values are considered.

**ii. Threshold Initialization 1**

The variable initial threshold is computed. This initial threshold ( was computed as a subtraction between the join mean (mean of all 9 pixels in step i) and low mean (mean of pixels denoting crack).

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

**iii. Final Threshold Computation**

The final threshold for each image was then determined using expression shown in Equ. 7.

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

In total two parameters, Kernel size, and division , were tuned.

**3.3 Denoising using Skele-Marker Method**

Binary image denoising is the process of removing unwanted artifacts and random variations in binary (black and white) images to reveal the true structural features of objects in the image. In binary images, each pixel is either entirely black or entirely white, with no grayscale values in between. Noise in binary images can obscure object boundaries and affect the accuracy of image analysis and processing. Denoising binary crack images is thus a crucial step in enhancing the accuracy of crack detection and analysis. The presence of noise complicates the identification and segmentation of cracks (Chianese et al., 2021). By addressing noise, there is better reliability and accuracy of crack detection algorithms. There are several methods that can be used for denoising, however, in the present study, the Skele-Marker was selected because it has been demonstrated to performance excellently well as evident in the literature (Dow et al., 2023). Table 2 shows the visual representation of the images at each step of denoising technique with respect to its implementation in this study with the python programming language which was not used in the original paper.

**Table 2** Visual of Skele-Marker Implementation

|  |  |
| --- | --- |
| **Step** | **Image** |
| Binary Image Post CCELD - |  |
| Area Thresholding – | Some Noise is removed |
| Skeletonization |  |
| Length Thresholding – | Most of the Noise is removed |
| Morphological Reconstruction 1 - | **End Crack is missed** |
| Drawing Circles at Endpoints – |  |
| Morphological Reconstruction 2 ­– | **Most of End Crack is Restored** |
| Final Image is derived |  |

**3.4 DL Segmentation Model (UNET)**

UNET is a popular architecture in the field of computer vision, specifically designed for semantic segmentation tasks. Semantic segmentation involves labeling each pixel in an image with a corresponding class label, making it a fundamental technique for tasks like object recognition and scene understanding. The UNET architecture is characterized by its U-shaped design, which consists of an encoding path and a decoding path.

**3.4. 1 Loss functions**

Every DL model is trained with a loss function which are means to update the weights of the model during training. There are several loss functions used such as the binary cross entropy loss, dice loss, focal loss, etc. The Binary Cross-Entropy Loss, uniquely tailored for binary segmentation tasks like crack segmentation, excels in penalizing models based on the probabilistic differences between predicted and actual crack pixels. In this study, the jaccard variant of the BCE loss is used and it is termed the BCE jaccard loss. The jaccard index or IoU calculates the overlap between predicted and ground truth masks by considering their intersection over their union. It's commonly used to evaluate the similarity or overlap between two sets.

The BCE Jaccard Loss expressed in Equ. 10 combines these two concepts by incorporating the binary cross-entropy loss to handle pixel-wise predictions and the Jaccard loss to measure the similarity between predicted and true masks. By doing so, it encourages the model to not only accurately predict pixel-wise probabilities but also to strive for better overlap and alignment between predicted and ground truth masks, ultimately aiming for improved segmentation accuracy and precise boundary delineation this makes it better for binary segmentation tasks over the other loss functions.

**The Binary Cross Entropy (BCE) Loss Function**

The binary cross entropy is a loss function that gives a score based on how far or close the probability of pixels in our training images are close to their true values (1 or 0 in the case of binary segmentation problems) (Udoh, 2023). For a flattened (1D) image mask, it is expressed as;

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

**Jaccard Loss**

The Jaccard loss is calculated as 1-IOU Score. For a flattened (1D) image mask, the loss function can be represented as;

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

The BCE Jaccard loss is a combination of both loss functions;

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

Where, = ground truth pixel value (1 or 0) and = predicted pixel value (probability)

**3.4.2 Model hyperparameters**

The VGG 19 was used as the encoder for the UNET model based on the literature (Liu & Wang, 2022). Towards implementation, the open-source python framework “segmentation models” built on top of Keras and Tensorflow is used (Iakubovskii, 2023). Towards optimal and accurate learning, the Adam optimization method is employed. Furthermore, the RGB image is resized to 256 × 256 × 3 and the ground truth is resized to 256 × 256 × 1 (binary image). A learning rate which was optimal at 0.01 after testing shown in section 4.4 was used. Finally, a mini-batch gradient descent approach was taken with a batch size of 32 images.

**3.5 Model Development and Use**

The training of the CNN model (mobilenet) was done using the python programming language and utilizing the Graphics Processing Unit (GPU) on Google Colaboratory while the grid search for optimal parameters for the image preprocessing technique was done across multiple computers with typical configuration on 4 cpu cores, core i5 and 8gbs of RAM as well as on Google colaboratory and on amazon sage maker studio labs for maximum speed. With thousands of cores, GPUs excel in handling computationally intensive operations like matrix multiplications crucial for neural network training. Their efficiency accelerates the training of complex models, reducing processing times from weeks to hours (*Google Colaboratory*, 2021.)