Hindawi
Advances in Civil Engineering
Volume 2020, Article ID 8848945, 26 pages
https://doi.org/10.1155/2020/8848945

Research Article

Implementation of Parallel K-Means Algorithm to Estimate Adhesion Failure in Warm Mix Asphalt

Mohammad Nishat Akhtar,[1] **Waseem Ahmed**,[2] **Muhammad Rafiq Kakar**,[3,4] **Elmi Abu Bakar**,[1] **A. R. Othman**,[5] **and Moises Bueno**[3]

[1]*School of Aerospace Engineering, Universiti Sains Malaysia, Nibong Tebal 14300, George Town, Penang, Malaysia*
[2]*Faculty of Computing and IT, King Abdulaziz University, Jeddah, Saudi Arabia*
[3]*Empa Swiss Federal Laboratories for Materials Science and Technology, CH, Dübendorf 8600, Switzerland*
[4]*Department of Architecture, Wood and Civil Engineering, Bern University of Applied Sciences (BFH), CH-3012 Bern, Switzerland*
[5]*Department of Mechanical Engineering, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Perak, Malaysia*

Correspondence should be addressed to Elmi Abu Bakar; meelmi@usm.my and A. R. Othman; rahim.othman@utp.edu.my

Received 6 May 2020; Revised 10 September 2020; Accepted 19 September 2020; Published 4 November 2020

Academic Editor: Tayfun Dede

Warm Mix Asphalt (WMA) and Hot Mix Asphalt (HMA) are prepared at lower temperatures, making it more susceptible to moisture damage, which eventually leads to stripping due to the adhesion failure. Moreover, the assessment of the adhesion failure depends on the expertise of the investigator's subjective visual assessment skills. Nowadays, image processing has gained popularity to address the inaccuracy of visual assessment. To attain high accuracy from image processing algorithms, the loss of pixels plays an essential role. In high-quality image samples, processing takes more execution time due to the greater resolution of the image. Therefore, the execution time of the image processing algorithm is also an essential aspect of quality. This manuscript proposes a parallel k means for image processing (PKIP) algorithm using multiprocessing and distributed computing to assess the adhesion failure in WMA and HMA samples subjected to three different moisture sensitivity tests (dry, one, and three freeze-thaw cycles) and fractured by indirect tensile test. For the proposed experiment, the number of clusters was chosen as ten ($k = 10$) based on k value and cost of k means function was computed to analyse the adhesion failure. The results showed that the PKIP algorithm decreases the execution time up to 30% to 46% if compared with the sequential k means algorithm when implemented using multiprocessing and distributed computing. In terms of results concerning adhesion failure, the WMA specimens subjected to a higher degree of moisture effect showed relatively lower adhesion failure compared to the Hot Mix Asphalt (HMA) samples when subjected to different levels of moisture sensitivity.

1. Introduction

The image processing method has been widely used as a nondestructive system to evaluate 2D or 3D geometry in numerous scientific fields [1]. In the field of civil engineering, image processing has been successfully applied in multiple applications such as pavement distress assessment, site evaluation using satellite imaging, and analysis of crack propagation and microstructures in cement-based materials [2]. Concerning asphalt pavement binders, a number of computer-vision based system has been developed, which is broadly categorized into field assessment and laboratory applications. Concerning field assessment, algorithms were developed for the identification of pavement distress type and size [2–5]. Nonetheless, these algorithms also allow identifying the fractured surfaces of asphalt mixtures such as broken aggregates or the adhesion and cohesion failures at the interface of failed specimens [1–3].

In this work, the type and degree of failure due to the moisture effect is evaluated by image analysis for a series of Warm Mix Asphalt (WMA) mixtures. WMA technology uses lower temperatures in comparison with conventional Hot Mix Asphalt (HMA). Even though in contrast with conventional Hot Mix Asphalt (HMA), the WMA

technologies are more environmentally friendly and cost-effective; however, due to the lower production temperature, the WMA mixtures have shown less resistance against moisture damage [6–8]. Concerning distress or stripping identification using a computer vision-based system, it is essential to know that the distressed/stripped area pixel always possesses a lower pixel intensity compared to the unstripped area [9]. Conventional tests carried out to identify moisture damage region of interest (ROI) in asphalt mixtures include the use of visual assessment depending on the perception of the investigator [10], [11]. In such a scenario, the investigator might miss out to select some part of the ROI, which finally hampers the end result. Therefore, to further understand the mechanisms, it seems necessary to use an image segmentation algorithm that can quantify these failures more accurately. It is deemed necessary to note that these types of image processing involve high-performance computation because the high-end image processing cannot be handled efficiently on single computing node. In this regard, a framework is needed that allows the researchers to concentrate on the image processing tasks and refrains by getting them involved into complicated details of distributed computing. Additionally, the framework should provide the researchers with the familiar image processing tools.

This work proposes an implementation of clustering on high-resolution images of WMA broken samples using a parallel k means for image processing (PKIP) algorithm. To implement the k means algorithm parallelly, multiprocessing on a single node and Map-Reduce based programming on multiple nodes have been used [12].

An essential factor in the visual assessment of the stripping is concerning to its representation. In terms of pixel value intensity, the stripped and the unstripped area must have a significant difference. In order to classify the pixels with respect to different intensities or color similarities, the k means clustering algorithm is widely used. Using the k means algorithm, n observations can be segregated into k classes based on a particular mean value. The proximity of each observation to the cluster is iteratively processed using the nearest mean. The variant of k means comprises k median and k medoid (see Appendix section). It is worth to be noted that k median is best suited for local optimization problem, whereas k means algorithm is tailored for both, i.e., global and local optimization [13]. On the other hand, k medoid has high algorithmic time complexity $o(n^2 * k * i)$ (see Appendix section), and thus, it is compute intensive compared to k means $o(n^2 * k * i)$ [14, 15].

After this short introduction, the manuscript is arranged as follows, Section 2 describes the previous work carried out on asphalt mixture analysis using conventional image processing techniques and also describes the challenges in terms of processing time required to analyse the high-end image datasets using sequential k means clustering. Section 3 starts with WMA specimens' preparation and proposes the PKIP algorithm for central processing unit- (CPU-) based multiprocessing execution and Map-Reduce based distributed computing, followed by a thresholding process to assess the adhesion failure. Section 4 presents the results in terms of the accuracy and performance of the proposed PKIP algorithm. Section 5 presents the discussion related to the outcome of the results. Finally, Section 6 summarizes the most significant conclusions.

2. Background

WMA technology was developed to place the asphalt mixture at a lower temperature compared to the conventional mixture, and, in recent years, WMA has gained prominence over conventional HMA approaches due to its sustainability factor. Nevertheless, due to the lower production temperature of WMA, the characteristics of probable moisture damage are more essential to be considered [7], [8]. In this sense, it is necessary to understand the effects occurring at the interface of the mineral aggregate and the asphalt binder. As per the observation of the National Center for Asphalt Technology (NCAT), the reduced mixing temperature used in the preparation of WMA leads to improper drying of the aggregates, and less aging takes place. Figure 1 shows the schematic diagram of moisture entrapment at the asphalt/aggregate interface during WMA production. Digital image processing has become a powerful tool to accurately accomplish the assessment of this type of damage [11].

Generally, image processing experiments are divided into destructive and nondestructive evaluations. In the case of destructive testing, experiments are ought to be carried out on specimen's failure to assess the performance of the material under different conditions by breaking it, whereas in nondestructive testing, the sample is kept intact during its analysis using digital image processing. In the work carried out by Li et al. [17], an artificial neural network was used to detect the cracks in the pavement whereby classification and preprocessing step was performed, followed by Gaussian filter utilization to smoothen the background. At the final step, histogram transformation was applied to highlight the region of crack. The results obtained by image processing were compared with the experimental parameters, which were obtained in lab, and an accuracy of approximately 80% was achieved [17]. Previous studies dealt with the modeling of the microstructure of the asphalt mixture also included sensitivity exploration of aggregate size inside sand mastic [18]. Image analysis is also used for the characterization of air void distribution in asphalt mixes utilizing X-ray Computed Tomography [19, 20]. Moreover, the 3D microstructure of the asphalt concrete can also be reconstructed from pieces of 2D X-ray computed tomography graphics to obtain the required macroscopic specification [21, 22].

Numerous studies and analysis present how to build up and enhance the image processing technique research-based commercial applications such as ImageJ, Image Processing, and Analysis (IPas) or Photo Pro Plus [20, 23, 24] to be implemented on specific domains. Generally, the sizes of modern image collections are large (terabytes and petabytes of data); such vast collections of data cannot be stored and processed efficiently on a single machine. In addition, current image processing algorithms are becoming very challenging and, hence, computationally intensive. There are

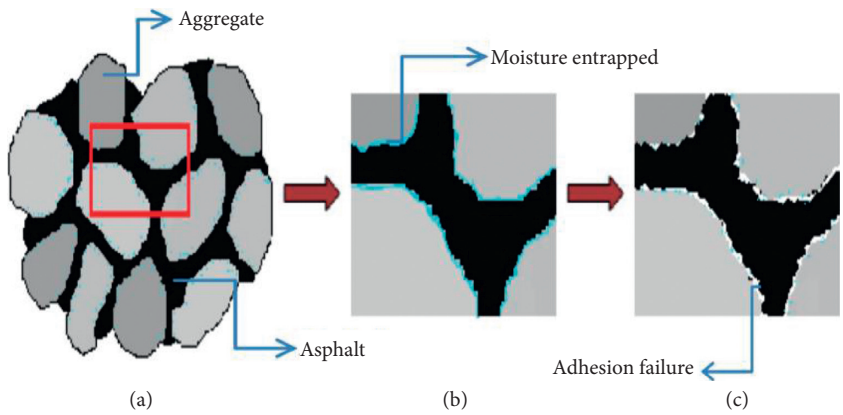


Figure 1: Moisture damage in warm mix asphalt holding moist aggregates (adopted from [16]) (a) Asphalt mixture. (b) Binder aggregate interface. (c) Damaged interface.

many compute intensive steps involved in processing large and complex image datasets to derive modified products, and several efforts are required towards integration of high-performance computing models with image processing algorithms. Although we can process the images batch-wise in these single-processor systems, there will be problems with such processing due to limited capabilities. Henceforth, a parallel programming paradigm for high-end image processing has become deemed necessary.

In image processing, the input image undergoes from several modifications [25, 26]. Implementation of image processing performs four steps: (i) image acquisition, (ii) image enhancement, (iii) image restoration, and (iv) multiresolution processing. In order for an image to be segmented and examined, initially it has to undergo a thresholding process. In image processing of asphalt sample, the image sample undergoes from several preprocessing methods focused on image smoothening to remove the noise from the image [27]. For the analysis of asphalt samples, acquisition of high-quality image is a necessary parameter and so is its intensive computation which is considerably time-consuming using sequential processing. In order to determine ROI in asphalt image sample, initially, the sample image has to be smoothened to segregate the object that determines fractured surface and the remaining unwanted background. For example, to implement this process of segregation, several image processing functions were applied by Dong et al. [28] to analyse the cracks present in asphalt samples. These image processing functions comprised of implementation of filter (i.e., Kalman, Gaussian, Sobel, polynomial, bilateral filter) to remove the noise and then the implementation of thresholding process with Canny edge detection [2] was done followed by morphological implementation operations to analyse contours. Thresholding on grayscale images were also applied to estimate stripping in untreated and treated asphalt mixture [2]. There are limitations of simple thresholding as it is a manual process and it does not resonate well with the dark color aggregate sample. Moreover, simple thresholding does not classify the shadow area accurately [29]. In order to overcome these shortcomings, Lantieri et al. [30], proposed a method of color space conversion, where they converted their sample image from Red-Green-Blue (RGB) color space to YUV color space (Y denotes luminance component, U and V are the chrominance color component).

The conventional image segmentation algorithm primarily contains the segmentation method depending on threshold value, the segmentation method based on the border, and the segmentation method based on the region [31, 32]. However, in order to determine the dominance of any pattern in the image, the k means clustering has turned out to be an efficient method compared to other clustering method as highlighted in Section 1. Cluster analysis is an important aspect for pattern/behavioral analysis [33]. Using k means algorithm, one can find out how to distinguish different kinds of elements by continuously enhancing the subliminal clustering pattern. Different clustering methods are continuously proposed and enhanced with the help of k means algorithm which are widely being used in medical image analysis using both CPU and graphical processing unit (GPU) computing [34]. The clustering process in k means algorithm is time-consuming and increases substantially with the increase in dataset size if implemented sequentially. However, due to the high efficiency of k means algorithm, it is widely utilized at the clustering of large-scale data using GPU [35, 36]. At present, many algorithms are extended and improved by keeping k means algorithm at the pivot. Compared with the traditional k means procedure, the adaptive k means algorithm used by Zheng et al. [29] transforms an image to the LAB color space before segmentation and places the luminance (l) into an adjusted value to reduce the disturbance resulting from the background. In one of the recent studies, sequential k means clustering and machine learning-based classification was used to estimate the stripping in asphalt coating [2]. It is essential to note that with respect to machine learning algorithm, the training process could be time-consuming. However, machine learning-based classification could be helpful to predict the stripping in asphalt mixture sample. It could be inferred from previous research that k means algorithm has the capability to be executed in parallel to cut down the execution time [37, 38]; therefore, a parallel implementation of k means algorithm can considerably enhance the execution time of image segmentation and

additionally optimize the algorithm structure to a specific extent on multiple cores. In order to estimate the average count of pixels associated to region of interest (ROI) on multiple image dataset, Akhtar et al. [39] implemented parallel image segmentation using Map-Reduce framework. In addition to this, Akhtar et al. [12], also made detailed analysis of execution pattern with respect to CPU run time and accuracy for the multiple input image dataset to be segmented in batch. It was evident from their results that higher-order dataset size scaled well with increasing number of compute nodes or compute cores.

In contrast to supervised learning, clustering is seen as an unsupervised learning method, as we do not have the ground truth data in comparing the output of the cluster algorithm with real labels to assess its performance. Numerous k means-based clustering algorithms use distance measurements to determine the similarity between data points [40–42]; however, it is advisable to normalize the data to have a zero average and standard deviation, as the characteristics in each data set would almost have different measurements. The k means algorithm is good at capturing data structures if clusters have a spherical shape [43, 44]. Moreover, in the field of artificial intelligence, k means clustering is used for hard clustering problems. Before discussing the methodology with respect to the k means clustering, Sections 3.1 and 3.2 give a brief description regarding WMA sample preparation and moisture conditioning.

3. Methodology

3.1. Asphalt Mixture Preparation. Conventional asphalt binder PG-76 was utilized in this study. In Table 1, the rheological properties of the PG-76 binder is summarized. Aggregate-type granite provided by Kuad Quarry Sdn. Bhd., Penang, was utilized in the preparation of the mixtures. The midrange aggregate gradation, type AC 14, used was in agreement with the Malaysian PWD gradation determinations [45, 46] for asphalt concrete. To evaluate the impacts of filler as anti-stripping agent in asphalt mixture, conventional Ordinary Portland Cement (OPC) and Pavement Modifier (PMD) were incorporated. PMD is a greyish-black powder mineral filler created in Malaysia which is utilized as anti-stripping agent. An incorporation of 5% of PMD by total weight of mixture, act as mineral filler in mixtures prepared [27].

Along with two HMA mixtures (with filler types OPC and PMD), WMA mixtures were prepared at different temperatures. The test specimens were compacted to the desired dimensions (height 63.5 mm and diameter 100 mm) by using the Servopac Gyratory Compactor. Air voids of the specimens for moisture sensitivity evaluation were kept at $7 \pm 1\%$. All the specimens were prepared according to the procedures defined by Asphalt Institute Manual (MS-2) [47].

In this study, the HMA mixtures taken as reference material were prepared at 180 degree Celsius and compacted at 170 degree Celsius. However, the WMA specimens were produced at a temperature of up to 30 degree Celsius lower than HMA as shown in Table 2 along with mixture designations.

The optimum binder content was determined as per the procedure of Marshall Mix design ASTM D1559 [48] which follows the the Malaysian Public Works Department (JKR) guidelines [46] for mix type AC-14. Moreover, a gyratory compactor was utilized corresponding to an assumed 30,000,000 equivalent single axle load [27]. Number of gyrations corresponding to initial compaction (N_{initial}), design compaction (N_{design}), and maximum compaction (N_{max}) was respectively 8, 100, and 125. In mixture design, the target air voids of (4 ± 0.1) % were used for all the specimens. The designed optimum binder content for both HMA and WMA were obtained as 5.2%. Moreover, for WMA, Cecabase® was used as warm mix additive at an application rate of 0.3% by mass of binder.

3.2. Moisture Conditioning and Assessment. On compacted asphalt specimens, moisture conditioning was performed to assess the effect of accelerated water conditioning followed by freeze-thaw cycles. For all compacted specimens, the conditioning was carried out as per the guidelines of ASTM D4867 [49]. The only exception in the guideline was the usage of distilled water with an addition of sodium carbonate at 6.662 grams concentration. This addition was used to make the pH level high in order to enhance the stripping rate/damage inside asphalt specimens [11]. The specimens were immersed in the solution and vacuumed for 15 minutes to achieve saturation levels between 55% and 80%. Afterwards, these specimens were exposed to freezing condition at (-18 ± 3) degree Celsius for 16 hours and thawing at 60 degree Celsius for 24 hours as one cycle according to ASTM D4867 [7, 49]. Three sets of specimens, unconditioned dry, conditioned after one freeze-thaw cycle (1FT), and conditioned after thee freeze thaw cycles (3FT), were separated.

The prepared specimens were then subjected to moisture sensitivity test according to American Society for Testing Materials (ASTM D4867) [7, 11, 45, 49]. The indirect tensile strength (ITS) test was used to assess the moisture susceptibility of mixtures at a test temperature of 20 degree Celsius as per the guidelines of ASTM, 2006.

After WMA sample preparation and moisture conditioning assessment, the next step was to develop a parallel image processing-based methodology to determine the adhesion failure in the WMA sample in the lowest possible execution time. For the proposed experiment, we have developed a parallel k means for image processing (PKIP) algorithm to perform the clustering on the WMA sample. Sections 3.3, 3.4, and 3.5 will illustrate the feature space, linearization model, and the implementation of the PKIP algorithm.

3.3. Features of the Input Image (Features Space). In image processing, a kernel function is used as a linear classifier to solve a nonlinear problem. For the classification of the kernel function-based algorithm, input space comprises of the original image, and feature space comprises the features of corresponding input image. The objective of applying the kernel-based k means clustering on the WMA specimens to distinguish between adhesion failure region and non-

Table 1: Properties of binder PG-76.

Property	Penetration at 25 degree celsius, 100 grams, 5 seconds (0.1 mm)	Softening point (celsius)	Ductility at 25 degree celsius (cm)	Flash and fire point (celsius)	Solubility (%)	Specific gravity
Test method	ASTM D5	ASTM D36	ASTM D113	ASTM D92	ASTM D2042	ASTM D70
PG 76	50	69	90	344	99.50	1.02

Table 2: Mixing and compaction temperature of asphalt mixtures.

Mixture type	Filler	Mixing temperature	Compaction temperature	Designation
HMA	PMD	180	170	HP180
	OPC	180	170	HO180
WMA	PMD	170	160	WP170
	OPC	170	160	WO170
	PMD	160	150	WP160
	OPC	160	150	WO160
	PMD	150	140	WP150
	OPC	150	140	WO150

adhesion failure region is to bring down the value of its expected square distance d (x) of the point from its center of cluster:

$$d(x) = \frac{1}{Q}\sum_{i=1}^{k}\sum_{p=1}^{n} x_p - c_i^2, \tag{1}$$

where $d(x)$ is the distance from the center of the cluster, Q is the site of the training set, k is the number of clusters, x_p is the current pattern, and C_i is the cluster center within the cluster i.

Now features of the input asphalt specimen $(\phi(x))$ could be computed from input space $X \in (x_1, x_2, x_3, \ldots, x_n)$ using

$$\phi(x) = \left(\phi_1(x), \phi_2(x), \phi_n(x)\right). \tag{2}$$

At the ground reality, the value of X is unknown. However, the inner product is known as kernel function (k) shown below:

$$k(x, y) = \langle \phi(x) \cdot \phi(y) \rangle, \tag{3}$$

where $\phi(x)$ and $\phi(y)$ represents the feature space in x and y direction.

3.4. Apply Kernel Function to Linearize the Model. Assuming in cluster C_p, the average features space $\phi(x)$ could be rewritten as

$$\phi_{\text{avg}}(X) = \frac{1}{C_p}\sum_{i=1}^{C_p} \phi(x_i). \tag{4}$$

Now there may exist a point which lies at the outer segment of $\phi(X)$ by appearing as an image to the input space.

In that condition $\phi(X)$ becomes inaccessible, but it is possible to compute its norm.

$$\begin{aligned}\phi_{\text{avg}}(X)^2 &= \left(\phi_{\text{avg}}(X_i), \phi_{\text{avg}}\left(X_j\right)\right) \\ &= \frac{1}{C_i^2}\sum_{i,j=1}^{C_i}\left(\phi(x_i), \phi\left(x_j\right)\right) = \frac{1}{C_i^2}\sum_{i,j=1}^{C_i} k,\end{aligned} \tag{5}$$

where $k = \phi(x_i), \phi(x_j)$.Now, assuming that, $\phi_{\text{avg}}(x) = \tau$.Therefore,

$$\tau^2 = \frac{1}{C_i^2}\sum_{i,j=1}^{c_i} k. \tag{6}$$

Then the distance d_{ij} when the input space is mapped to the features space could be computed as

$$\begin{aligned}\phi(x) - \tau^2 &= d_{ij}^2 \\ &= \left(\phi(x_i),\ \phi\left(x_j\right)\right) - 2\tau\phi(x) + (\tau \cdot \tau) \\ &= k\left(x_i, x_j\right) - 2\tau\phi(x) + \left(\tau_i, \tau_j\right).\end{aligned} \tag{7}$$

It is possible to kernelize the function in equation (7) by using a median filter given by the function: $k'(m, n) =$ median $\{x[i, j], (i, j)\, \varepsilon W\}$ (median filter function) where W represents a neighborhood defined by the user which is centered around location (m, n) in the image and $x[i, j]$ is 2D array of pixels comprising of i^{th} row and j^{th} column. In addition to the aforementioned filters, there are other filters as well, i.e., Gaussian filter, Bilateral blur filter, and Sigmoid filter which may be used as per the applications requirement [50, 51]. For the proposed experiment, the median filter has been used to kernelize the equation (7).

3.5. Implementation of PKIP Clustering Algorithm. This section intends to highlight the execution of a streamlined sequential and multicore CPU variant of the k means clustering algorithm. Therefore, the proposed algorithm was customized using multiprocessing programming to obtain comparative outcomes between parallel and sequential execution. The algorithm was tried with different asphalt

mixture image dataset. The image datasets were acquired using a Nikon D800 model. The dimensions of the image samples were 5520 pixels (length) and 3680 pixelsl (width), and the color space of each sample was maintained in RGB prethresholding process. It is deemed necessary to detail out the system configuration in terms of number of logical cores, threading mechanism, and memory on which the PKIP algorithm has to be implemented. The work relied on OS *X* 10.1 64-bit operating system with a dual-core i5 (Hyperthreading support) at 1.8 GHz with a turbo boost of up to 2.9 GHz having 3 MB shared L3 cache and 8 GB RAM. The proposed PKIP implementation was tested using 4 threads. The version of the C++ compiler used was 4.6.4, while the version of OpenCV (image processing library) used was 3.4.2.

A clear sequential execution was initially performed using the algorithm shown in Pseudocode 1; however, this could not be viewed as an ideal execution on the grounds that the algorithm had a few matrix-like relations which were basic for getting better execution. The matrix computation was involved in the kernelization phase and in the distance computation of each pattern with respect to center of cluster as shown in the Pseudocode 1. In order to get substantial speedup, iterative matrix multiplications could be implemented in a computer environment where it could be executed in parallel.

In the proposed algorithm shown in Pseudocode 1, phase no. 5 holds the key to our algorithm. Phase number 5 could be categorized into two stages, stage I and stage II, using which substantial speedup is obtained for the proposed algorithm. Stage I involves the kernelization phase, and stage II involves an iterative procedure to compute the distance of every pattern to the different center of clusters. Stage I includes registering of the kernel matrix, which uses the kernel function that is connected for each matching pattern. This calculation is similar to matrix multiplication with the exception that for the proposed experiment, several task operations are involved rather than basic matrix multiplication. Moreover, these task operations require dependencies on the utilized kernel functions comprising of subtraction, squaring, and division of the median filter kernel function (Pseudocode 1).

The distance computation between each pattern with respect to the cluster center comprises of three terms as

$$\text{distance}\,(i,j) = k'k(x,x) - \frac{2k'}{c_i}\sum_{m=1}^{n} L_{mj}\cdot\tau\phi(x) + \frac{k'}{c_i^2}\sum_{m=1}^{N}\sum_{n=1}^{N} L_{mj}L_{nj}\times\left(\tau_i\tau_j\right). \tag{8}$$

For equation (8), the first term is computed in Stage I. The computation of Stage I comprises of Phase no. (1) to Phase no. (4). The second and third term were computed in stage II. The second term is a direct multiplication of matrices, i.e., L and $\phi(x)$ followed by multiplication by $(2k'/c_i)$. The third term in the above equation depends upon the number of cluster and is continuously computed for every cluster. Since the third term is an iterative procedure and requires immense matrix multiplication, it was parallelized using multiprocessing programming. As a result, for equation (8), serial execution was exploited to compute the first term, whereas the second and the third term was implemented parallely using multiprocessing as it involved higher degree of iterative matrix multiplication.

In order to assess the performance of the PKIP algorithm for the proposed experiment, we have implemented it using C++ programming language by importing Open Computer Vision (OpenCV) library, which is used for image processing functions. The sequential execution of k means clustering can be further parallelized on multicore CPU. There are numerous application programming interfaces (API) available for parallel processing; however, for the proposed experiment, we have used Open Multi-Processing (OpenMP) for parallelizing the execution. OpenMP is an API for multiprocessing programming which supports shared memory architecture [52]. The prime feature of OpenMP is its vast instruction set i.e., OpenMP pragmas which is used for auto parallelization. For the proposed experiment, OpenMP pragma *omp for* was used to parallelize the intensive matrix computation involved in stage II. In order to divide the execution of Stage I between multiple cores, OpenMP *shared* construct was used.

3.6. Parallelization of K-Means Using Map-Reduce. In order to parallelize the k means clustering algorithm using Map-Reduce program, Hadoop framework with Hadoop Interface for Image Processing (HIPI) was used. Hadoop is an Apache open source framework written using java programming language by enabling distributed processing of large datasets across clusters of computer nodes using simple programming models [53]. The environment in which Hadoop works comprises of distributed storage and computation across clusters of computer nodes. Hadoop is designed to scale up from single node to thousands of nodes, of which each node offers local computation and storage.

Maps are the individual tasks which converts input records into an intermediate record. Any given input pair may map towards a minimum of zero or a maximum of n output pair. A Hadoop Map-Reduce structure spawn single map task for each input split was produced by the Inputformat for the job. The Reduce function of Hadoop framework is the second phase. Reducer minimizes the set of intermediate values passed by mapper and shares a similar key. The quantity of reduces for the job is set by the client by means of the reduce task function. In general, Reducer executions are passed the job configuration for the job through the job configurable class. The framework then calls the function reduce () for each key-value pair for the gathered inputs.

The Map function operates on every point x on the given image dataset. On a given point x, the squared distance between x and every mean is calculated and subsequently the mean M_i is determined which minimizes this distance. On the basis of these parameters, a key-value pair gets emitted

Input sample
k = number of cluster $(c_1, c_2, \ldots, c_i)$
Kernel function: $k(x, y) \leftarrow \phi(x_i), \phi(x_j)$
Phase no
(1) Apply kernel function to transform the input sample $k' = (1/(k_{\text{size}} \cdot \text{width} \times k_{\text{size}} \cdot \text{height})) \times \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix}$,
(2) Compute distance matrix $d_{i,j}^2 = \|\phi(x) - \tau\|^2$
(3) List pattern to the cluster randomly using label matrix, $L_{i,j}$ $L_{i,j} = f(x) = \begin{cases} 1, & \text{if } L_i \text{ in cluster } j, \\ 0, & \text{otherwise.} \end{cases}$
(4) Change $\leftarrow 0$
(5) Distance computation b/w each pattern w.r.t center of cluster)
while change $\neq 0$ do
for $j = 1$ to k do
compute cluster size c_i
for $i = 1$ to N do
compute distance of each pattern from the cluster using:
$k'k(x_i, x_j) - 2k'/c_i \sum_{m=1}^{n} L_{mj} \cdot \tau\phi(x) + k'/c_i^2 \sum_{m=1}^{N} \sum_{n=1}^{N} L_{mj} L_{nj} \times (\tau_i \tau_j)$.
end **for**
end **for**
Previous $L = L$
update label L
L = min. *val* of d along column
if Previous $L = L$ **then**
change = 0
end **if**
end **while**

Pseudocode 1: k-means using sequential execution.

with an index of mean "i" as key and $(x, 1)$ as the value. Therefore, a map function could be framed as

$$k \text{ means map } (\text{x}) : \text{compute} \left(\text{argmin}_i \left\| x - M_i \right\|^2, (\text{x}, 1) \right). \tag{9}$$

On the other hand, the Reduce function is just a pairwise summation performed on the values associated to each key. For instance, if two value pairs $[(x, q), (y, r)]$ are associated to a particular key, then a combined formation is possible by adding all elements in the given pair. In this regard, k means reduce function could be formed as follows:

$$k \text{ means Reduce } (\text{i}, [(x, q), (y, r)]) : \text{return } (\text{i}, (\text{xx} + \text{yy}, \text{qq} + \text{rr})). \tag{10}$$

Set of k values could be formed using the Map-Reduce characteristic of two functions shown as

$$\left(i, \left(\sum_{x \in P_i} x, \left| P_i \right| \right) \right), \tag{11}$$

where P_i denotes the set of points closest to mean M_i. Now the updated means could be computed as

$$\text{Mi} \longleftarrow \frac{1}{\left| P_i \right|} \sum_{x \in P_i} x. \tag{12}$$

For the proposed experiment, initial cluster centres (means) were determined using the canopy algorithm [54] whereby a set of initial distance threshold, i.e., T1 and T2 were assigned for each sample image such that T1 > T2. Post threshold initialization, the mapper function selects a sample vector randomly from the sample image and assigns it as a central vector of the canopy, and accordingly, it traverses the entire sample image. The distance between the scene image and the canopy central vector has to be less than T1 in order to get classified as canopy. If it is less than T2, then it gets discarded from the dataset. Finally, the output of the mapper function gets processed by the reducer to integrate the central vectors of the canopy. This generates a new canopy of the central vector which is considered as the initial clustering center.

In order to calculate the distance between a point x and each of the means using Map function, every compute node must possess set of current values of means. Therefore, in this regard, new means are circulated to all the compute nodes after the completion of each iteration. If the convergence is achieved after a specific iteration, then the execution gets completed; else, the new means/centroid are computed again using iterate () function.

Pseudocode 2 shows the implementation of k means using Map-Reduce.

Our experimental setup consists of 4 machines comprising of 1 master and 3 slaves. The master node is responsible to take the input image files. For the proposed experiment, Hadoop (version 2.7.1) framework was used in order to implement the parallel computation using 4 nodes. Figure 2 shows the setup configuration. The setting of Hadoop parameters was amended as shown in Table 3. Table 4 shows the specs of the nodes.

(1) Choose k initial value for the input image.
(2) Apply Map-Reduce $\forall$ k.
iterate (point, centroids)
{Assign points => clustersmap {(point => (point nearest to centroids) => (point, 1), (point n, count n))}reduce {((point1, count1), (point2, count2)) => (point1 + point2, count1 + count2)}Total_mapValues {(point_Total, count_Total) => point_Total/count_Total}
(3) Record the new means (Updated Centroid) post Map-Reduce.
(4) Circulate the updated centroid to each node in the cluster.
(5) **If** (converged (centroids, Updated_Centroids))
Then finish execution
Else iterate (points, Updated_Centroids)

Pseudocode 2: k means using Map-Reduce.

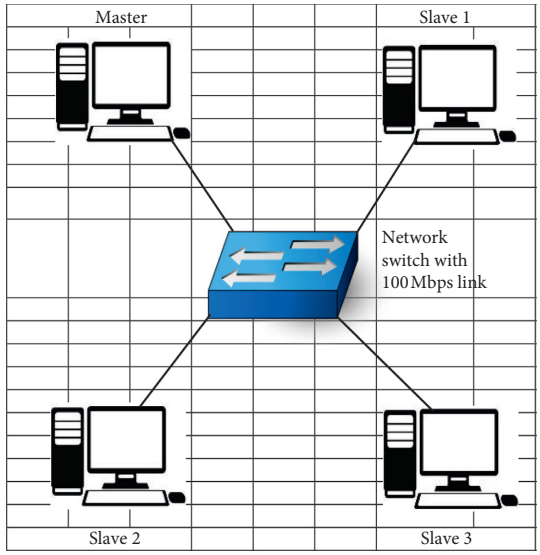


Figure 2: Experimental setup of Hadoop nodes.

Table 3: Configuration of Hadoop parameters.

Block size (MB)	Number of task mapping per node	Number of task reducing per node	Replication factor	Scheduler configuration	k value
128	2	1	2	First-in-first-out (FIFO)	10

Table 4: Configuration of nodes.

Nodes	RAM (GB)	Processor (GHz)	Dedicated hard disk drive (HDD) (GB)	No. of cores	Operating system
Master Slave 1 Slave 2 Slave 3	8	1.8	50	2	OS X 10.1

3.7. Thresholding of Processed Samples. For the analysis of asphalt mixture specimens post k means clustering, HSV thresholding was used (see Appendix section) [55].

In the proposed experiment, background and foreground segregation was significant to gain clear visibility of the ROI. For instance, the HSV color space for the boundary condition {[0, 1], [0, 1], [0, 255]} belongs to {$\text{Hue}_{\text{input}}$, $\text{Sat}_{\text{input}}$, $\text{Val}_{\text{input}}$}, the region of interest (ROI) being a specific tuned color object. Then the computation of the threshold pairs, i.e., ($\text{Hue}_{\text{lower}}$, $\text{Hue}_{\text{upper}}$), ($\text{Sat}_{\text{lower}}$, $\text{Sat}_{\text{upper}}$), and ($\text{Val}_{\text{lower}}$, $\text{Val}_{\text{upper}}$) could be utilized to convert the HSV color space image to the binary form using the below equation:

$$C(x, y) = \begin{cases} 1, \text{Hue}_{\text{lower}} < \text{Hue}_{\text{input}}(x, y) < \text{Hue}_{\text{upper}}, \\ \text{Sat}_{\text{lower}} < \text{Sat}_{\text{input}}(x, y) < \text{Sat}_{\text{upper}}, \\ \text{Val}_{\text{lower}} \le \text{Val}_{\text{input}}(x, y) \le \text{Val}_{\text{upper}}, \\ 0, \text{Otherwise}. \end{cases} \tag{13}$$

Here $C(x, y)$ is the segmented part. The equation (13) illustrates that if the HSV values for the pixels of the input image lies within the range of lower bound to upper bound values, then its associated output pixel belongs to class object 1, otherwise it gets designated to null (0) [56].

The flowchart in Figure 3 illustrates the HSV thresholding along with computation of zero and nonzero pixels. Figure 4 shows different images of specimens from the WMA dataset (Dry, 1FT, and 3FT) and Figure 5 shows its raw RGB model whereby it could be observed that the variable intensities of the brown component represents the adhesion failure in WMA samples. Figure 6 shows the different images of specimens from the HMA dataset and Figure 7 shows its associated raw RGB model. From the visual assessment point of view, the white region within the sample represents broken aggregates. If the RGB model is observed minutely for WMA and HMA samples, then a slight difference could be observed with respect to the brown component pattern for WMA and HMA samples, i.e., higher degree of the brown component is visible for HMA samples if compared to WMA for all the three categories, i.e., Dry, 1FT, and 3FT. (Figures 4–7).

4. Results and Discussion

The proposed work highlights the efficiency of PKIP algorithm for CPU-based execution by testing several datasets of asphalt mixtures. Analysis was done for the performance of proposed algorithms in terms of execution time and accuracy. For this study, three categories of dataset were evaluated, namely, dry, 1FT, and 3FT. At the same time, the influence of the mixing temperature (i.e., 180°C for HMA and 170°C, 160°C and 150°C for WMA) and type of filler (OPC and PMD) were analysed as well. The details of the different datasets are shown in Table 5.

In order to remove the noise, median filter was applied, and subsequently, the images of WMA along with its RGB color model and HMA samples along with its RGB color model is shown in Figures 8, 9, 10, and 11. After having a clear observation of the filtered images and its RGB color model, it could be seen that substantial noise elements have been blurred and the sample images has been polished. Post the application of median filter, the process of k means clustering becomes fine-tuned and thus helps in providing the accurate results.

Before initiating the explanation of CPU-based performance, it is required to highlight the accuracy of its execution timing result. With respect to execution time, it is important to note that there were fluctuations in the reading after each run of PKIP algorithm for every sample; therefore, the recorded errors were the best-obtained ones post four runs of the proposed algorithm. This performance comparison is shown in Figure 12. The fluctuation in the obtained values was within the range of |5%| from the best recorded value. Moreover, it is necessary to clarify that this error is due to the communication overhead among the threads (virtual version of CPU core) which results is extra waiting time.

As mentioned earlier in Section 3.5, with respect to the PKIP, the algorithm constitutes two stages, i.e., stage I involves the kernelization phase and stage II involves iterative procedure to compute the distance of every pattern to the different center of clusters. As stage II is an iterative process, it has to complete a predefined maximum number of iterations to attain the convergence. With respect to the proposed experiment, for each value of k, the cost of k means function was computed using vegas-viz plot library for Scala (see Appendix section) [57]. The best choice for k was chosen by observing the value which got minimized with little return gain. For the proposed PKIP algorithm, the best segregation was obtained by setting the cluster size to 10. Figure 13 shows one of the specimens (WP170 subjected to 3FT) with varying number of clusters (k). It is observed from each clustering stage that the different intensities of brown color (adhesion failure) are getting normalized. By keeping the value of $k = 10$, a fine segregation is achieved with respect to different intensities of color component which makes it a favorable factor for HSV thresholding. For the proposed experiment, the noise was removed using median filter function. From the visual assessment also, it is observed from Figure 13 that post cluster 5, maximum degree of segregation is achieved, or the intra-cluster variation has reduced within the sample region. However, we have chosen $k = 10$ as per the k value and cost of k means function as the best case.

Figure 14 shows the RGB color model along with histogram in RGB and HSV color space for the sample input image. In the RGB model, the encircled portion represents the adhesion failure. It is important to note that before initiating the clustering process, histogram equalization was applied on input image to maintain the uniform luminosity. It is observed from the histogram that most of the variation in the RGB components corresponds to a uniform lower range value of the pixel (low luminosity). However, at the extreme right of the histogram, higher peaks of RGB is to be seen, which clearly signifies clustered noise with high luminosity. It is also essential to note that the more the luminosity is, the higher the v value in HSV histogram is.

4.1. Discussion on CPU Implementation. In regard to the CPU implementation of the proposed PKIP algorithm, several ideas are noteworthy, especially with respect to the parallel code which leverages on the number of CPU thread(s) which is considered to be a virtual version of CPU core. Figure 15 shows the effect of incrementing the threads to execute PKIP clustering on different asphalt mixture datasets for $k = 10$. As shown in Figure 15, increasing the number of threads from 1 to 2 brings down the computational time in a nonlinear manner. This is due to the fact that all stages of computation involved in equation (8) do not scale directly with the available cores. For equation (8), stage I which does not possess iterative computation showed higher degree of linear scaling compared to stage II. This particularly signifies that all the components of stage II were not completely parallelized, and there was some composition of serial execution remaining in it. It is evident from the

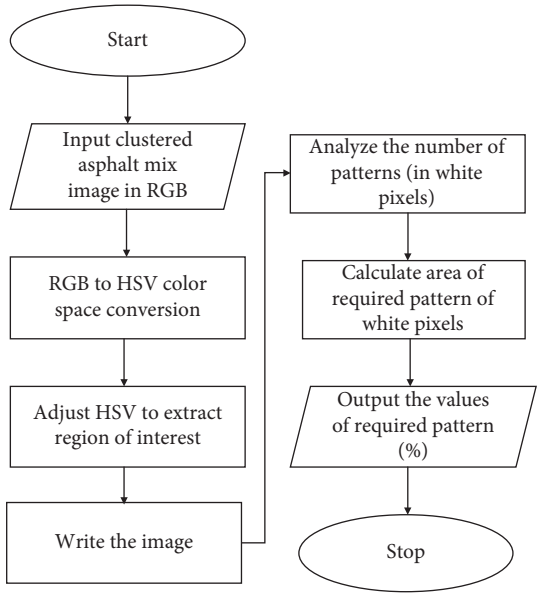


Figure 3: Process of image thresholding.

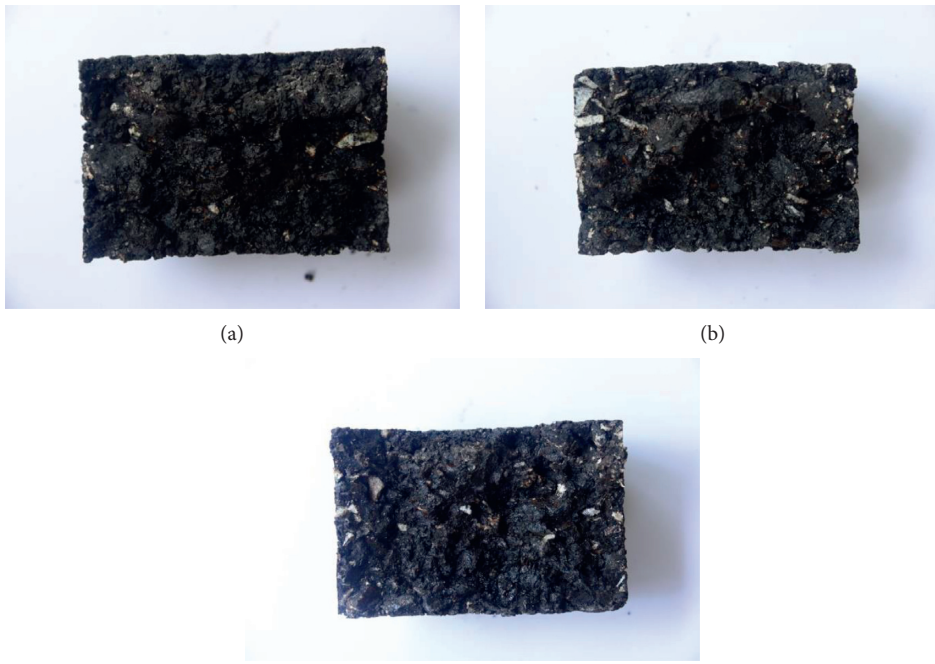
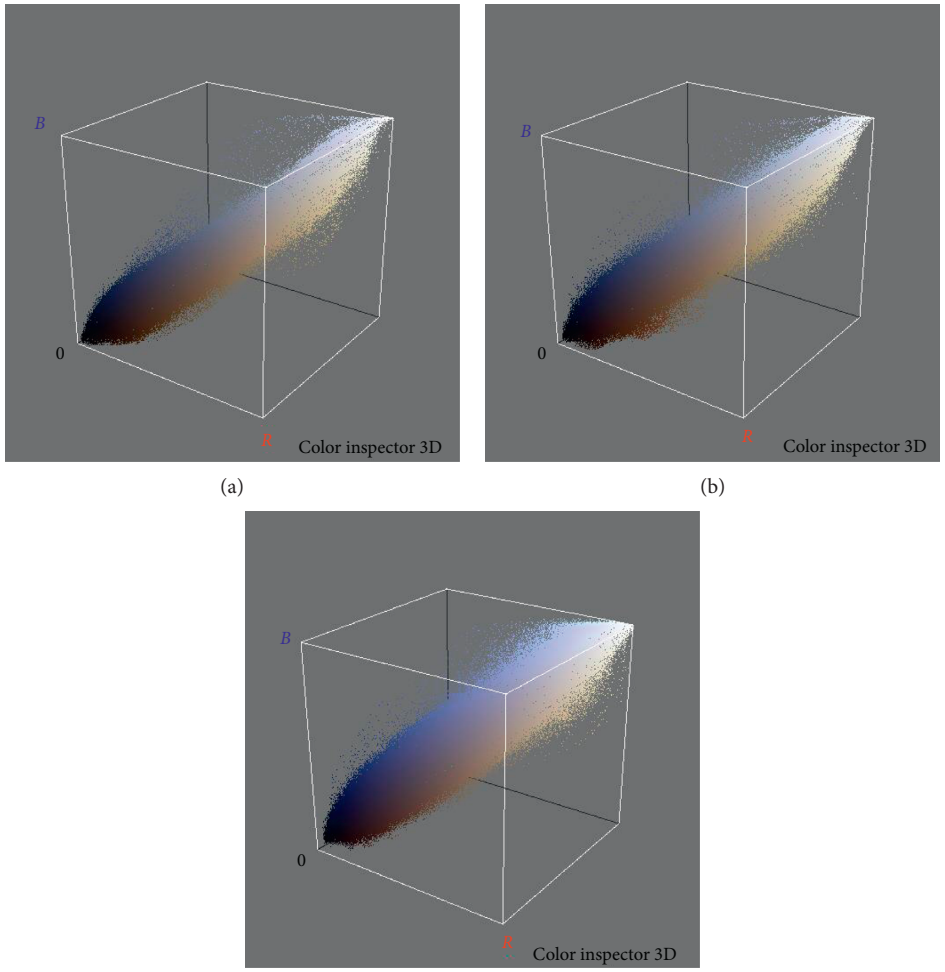


Figure 4: Sample specimen of (a) Dry-WP170, (b) 1FT-WO170, (c) 3FT-WP170.

execution time of C program (serial execution) and C + OpenMP (see Appendix section) program (parallel execution) that there is an approximately 25% to 30% improvement in terms of execution time with respect to C + OpenMP program, i.e., parallel execution of k means clustering.

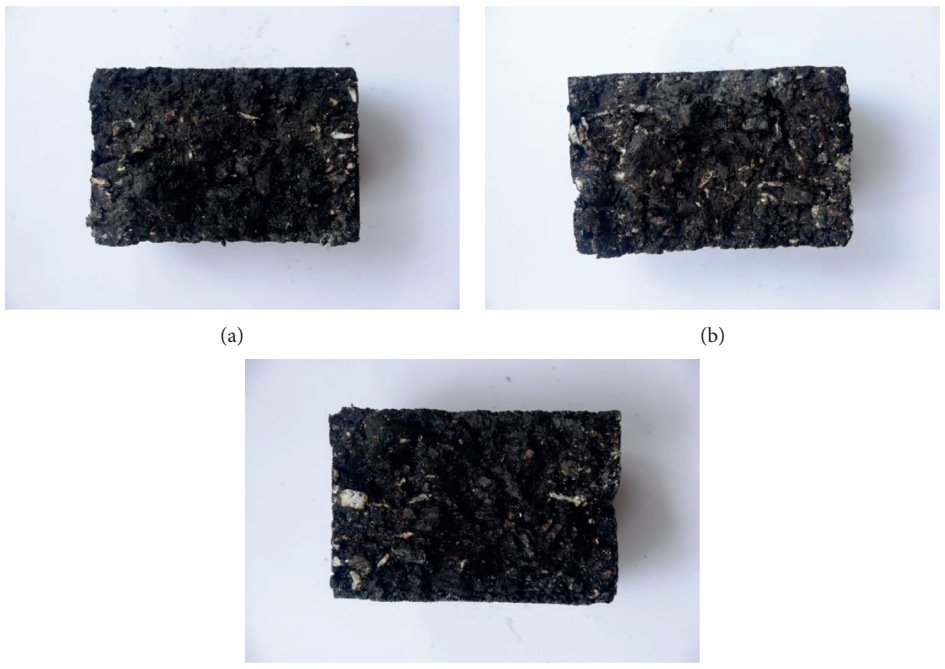
Moreover, when the number of threads is increased from 2 to 3, then it could be observed from the execution timing results that there is not much difference between single thread and triple thread results. In addition to this, when the number of threads was increased from 3 to 4, then there was an approximately 25% to 30% increment in the execution time of C + OpenMP. Therefore, it is evident from the obtained results that increasing the number of threads does not always increase the execution time subjected to the specification of the CPU. With context to effect of hyperthreading, refer to Appendix section.

Tables 6, 7, and 8 show the execution timing (sec) results of PKIP algorithm implemented both sequentially and parallelly on the WMA and HMA samples. Table 9 shows the



(a) (b) (c)

Figure 5: RGB-model of (a) Dry-WP170, (b) 1FT-WO170, (c) 3FT-WP170.



(a) (b) (c)

Figure 6: Sample specimen of (a) Dry-HP180, (b) 1FT-HO180, (c) 3FT-HP180.

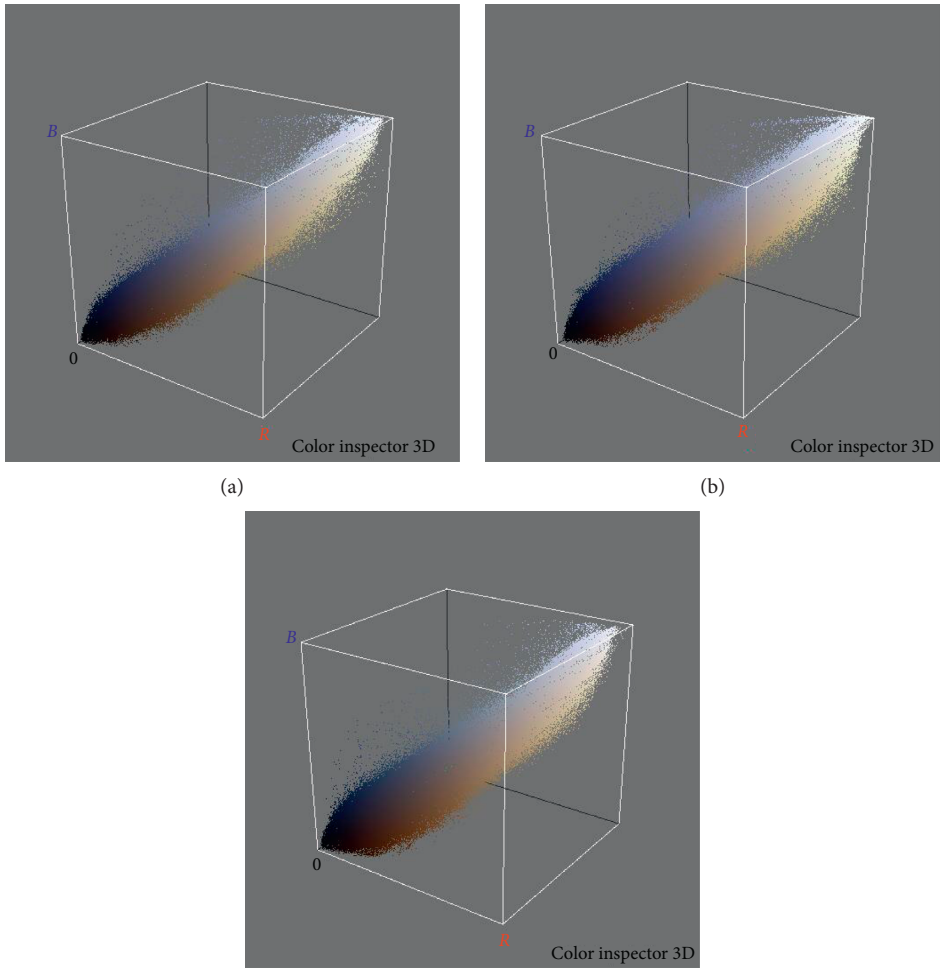


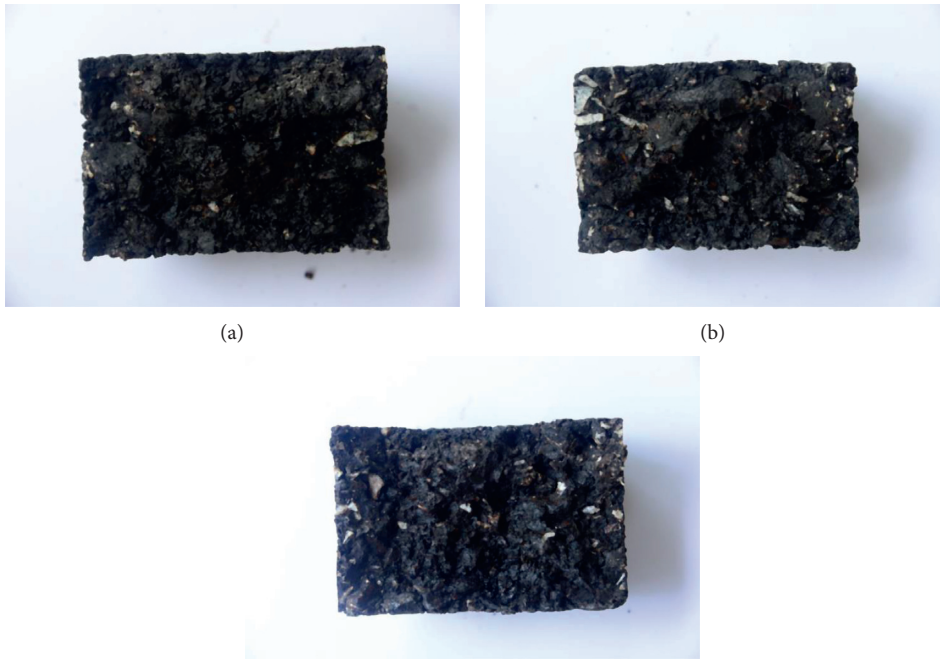
Figure 7: RGB-model of (a) Dry-HP180, (b) 1FT-HO180, (c) 3FT-HP180.

Table 5: Datasets of asphalt mixture samples.

Datasets					
Dry		1FT		3FT	
Sample	Sample code	Sample	Sample code	Sample	Sample code
WP170	1	WP170	9	WP170	17
HP180	2	WP160	10	WP150	18
HO180	3	WO170	11	WO170	19
WO150	4	WO150	12	WO150	20
WO160	5	WP160	13	WP160	21
WO170	6	WO160	14	WO160	22
WP160	7	HO180	15	HO180	23
WP150	8	HP180	16	HP180	24

gain in speedup between sequential implementation using C program (Serial Execution) and the parallel implementation using C + OpenMP for $k = 10$.

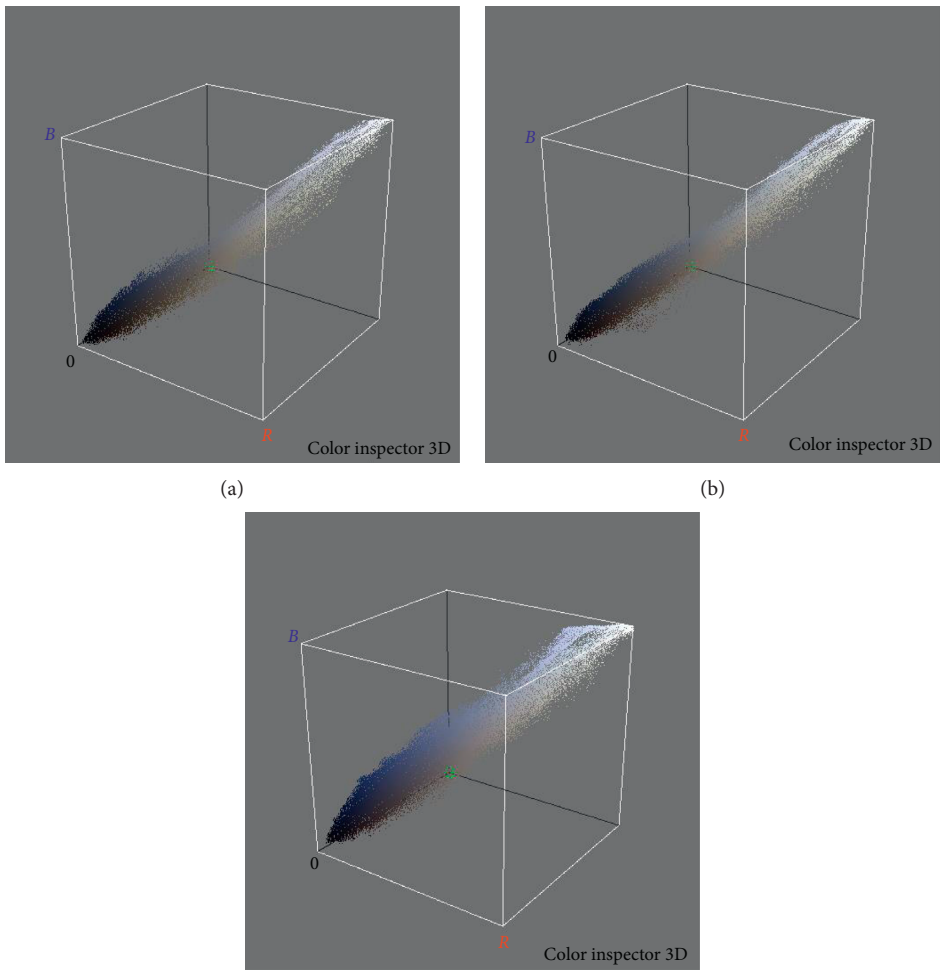
It is inferred from Tables 6, 7 and 8 that when the number of clusters (k) increases then PKIP algorithm scales well if compared with the serial or sequential programming for all the datasets. In particular, the best timing difference was obtained for $k = 2$, between parallel k means and sequential. It can be observed that for dry conditions, a wider gap by considering lower execution time for C + OpenMP is 9.49 sec, similarly for $k = 4$, 6, 8, and 10, the associated timing difference is 14.84 sec, 22.30 sec, 30.73 sec, and 37.17 sec, respectively. For 1FT condition specimens, the best timing difference for $k = 2$, between parallel k means and sequential k means is 9.43 sec while for $k = 4$, 6, and 10 the associated timing difference is 18.87 sec, 28.43 sec, 38.34 sec, and 47.47 sec, respectively. Finally, for the specimens conditioned with



(a) (b)

(c)

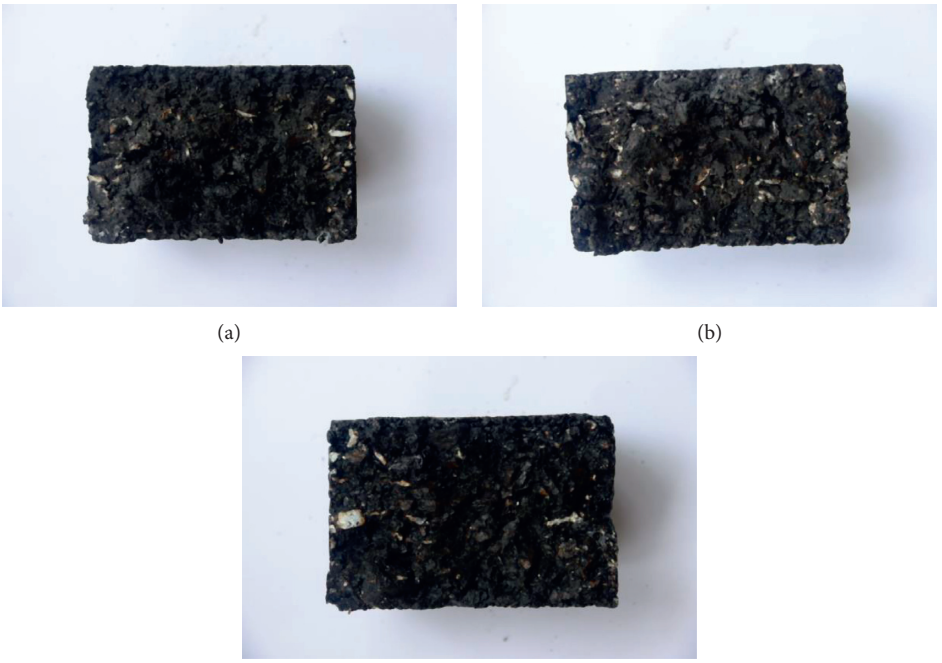
Figure 8: Post median filter-sample specimen of WMA. (a) Dry-WP170, (b) 1FT-WO170, (c) 3FT-WP170.



(a) (b)

(c)

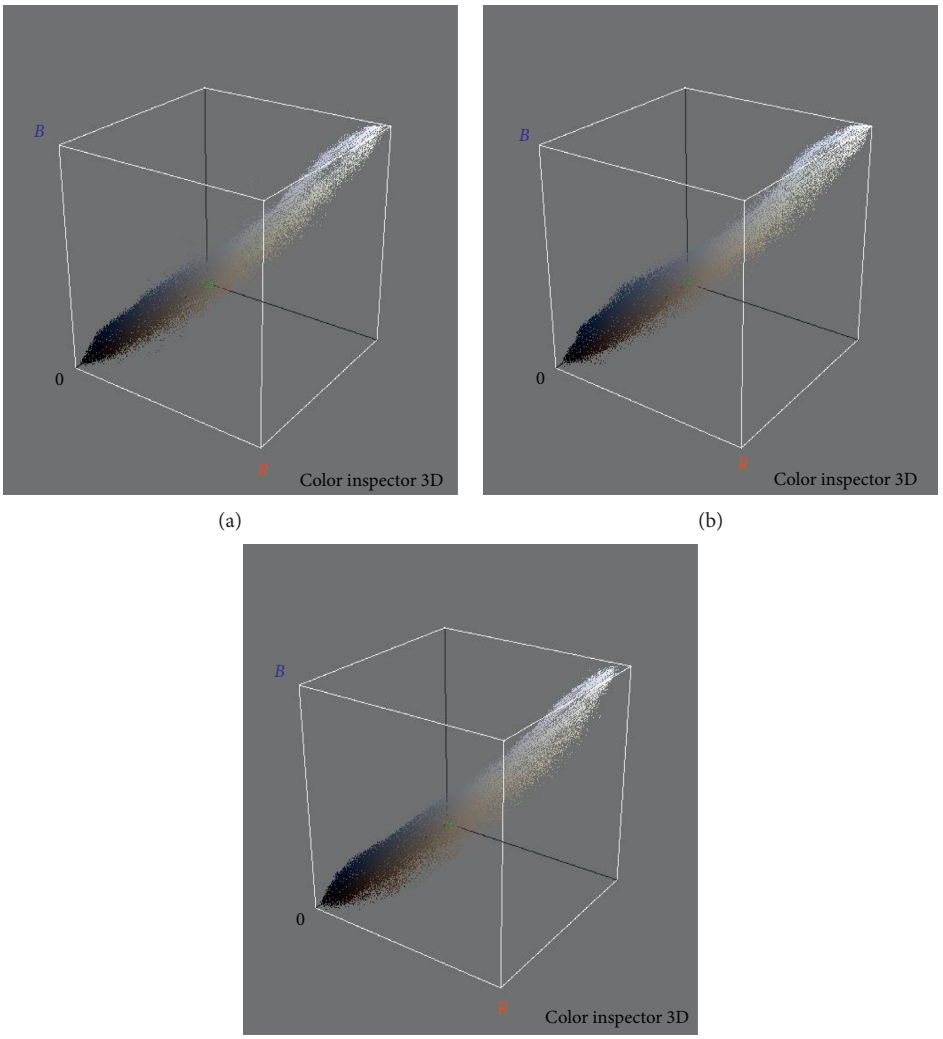
Figure 9: RGB-Model of median filtered. (a) Dry-WP170, (b) 1FT-WO170, (c) 3FT-WP170.



(a) (b)

(c)

Figure 10: Post median filter-sample specimen of HMA. (a) Dry-HP180, (b) 1FT-HO180, (c) 3FT-HP180.



(a) (b)

(c)

Figure 11: RGB-Model of median filtered. (a) Dry-HP180, (b) 1FT-HO180, (c) 3FT-HP180.

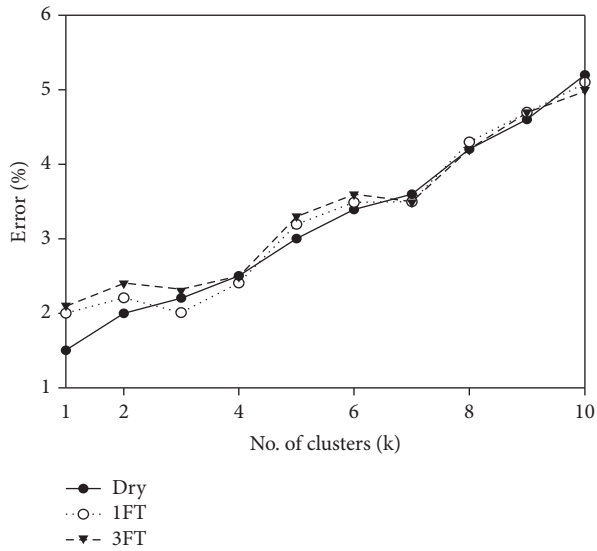


Figure 12: Accuracy of PKIP using openMP.

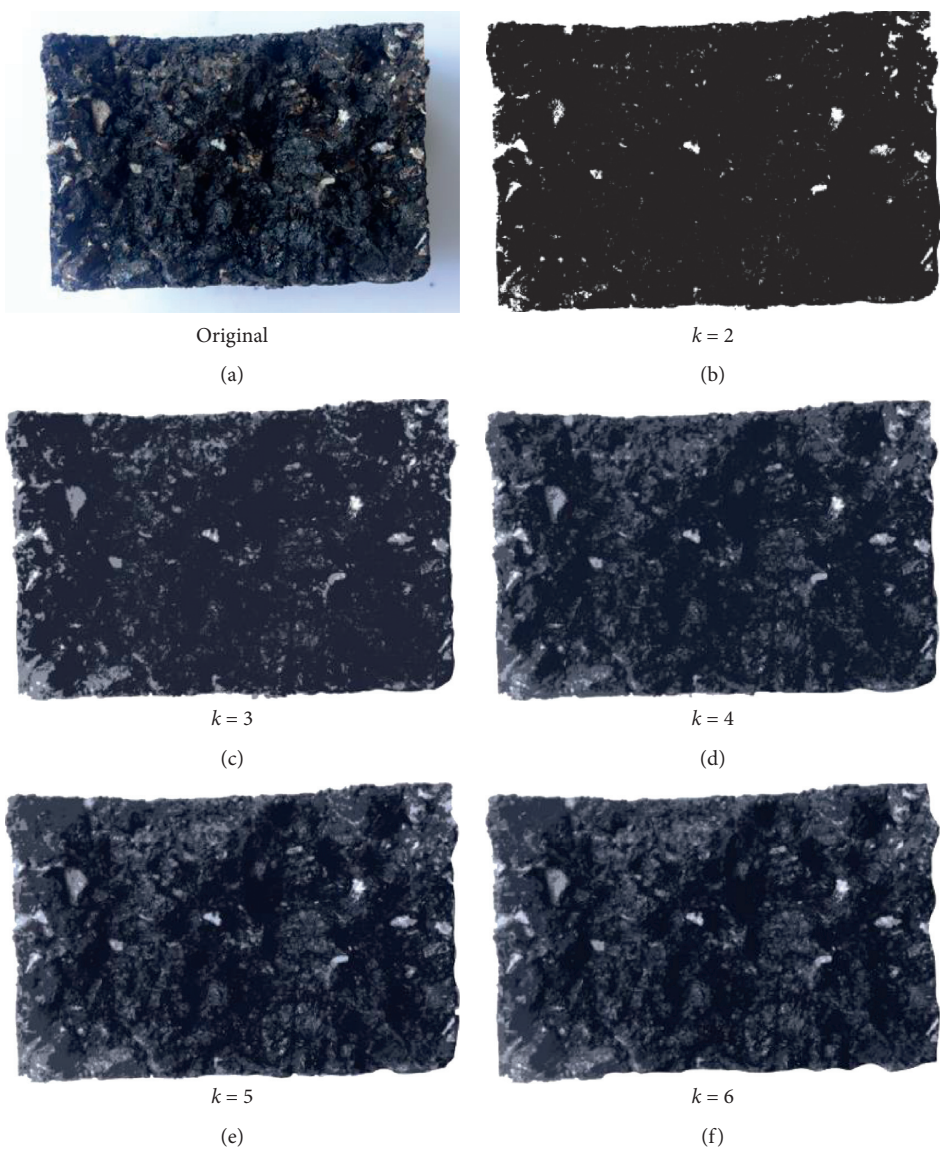


Figure 13: Continued.

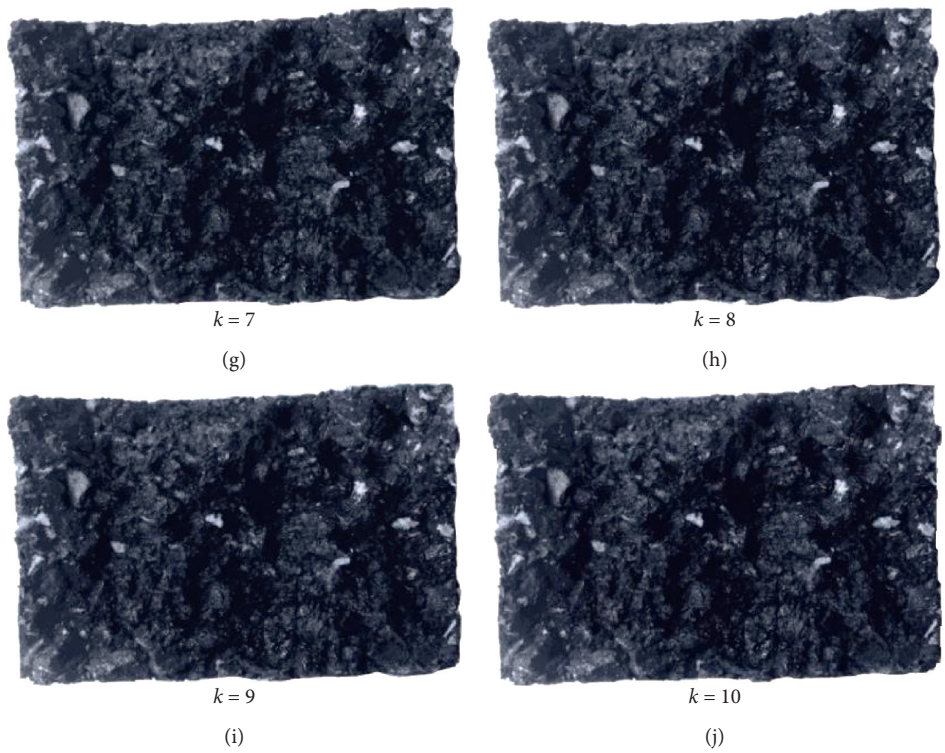


Figure 13: Clustered images of WP170 subjected to 3FT sample specimen. (a) Original. (b) $k = 2$. (c) $k = 3$. (d) $k = 4$. (e) $k = 5$. (f) $k = 6$. (g) $k = 7$. (h) $k = 8$. (i) $k = 9$. (j) $k = 10$.

3FT, the best timing difference for $k = 2$, between parallel k means and sequential k means is 9.03 sec, whereas for $k = 4$, 6, 8 and 10 the associated timing difference is 18.07 sec, 22.34 sec, 31.66 sec, and 45.18 sec, respectively.

4.2. Map-Reduce-Based Implementation. Tables 10, 11, and 12 compare the execution timing result of Map-Reduce-based implementation of PKIP algorithm over single node and multiple nodes for Dry, 1FT, and 3FT image datasets. Table 13 shows the speedup comparison of Map-Reduce-based PKIP algorithm implementation with its sequential implementation using C-program and parallel implementation using C + OpenMP. It could be inferred from the results that the Map-Reduce based PKIP algorithm scales well with the increase in nodes. For the execution with respect to single-node implementation of PKIP algorithm, an overhead resulted post splitting of algorithm in Map and Reduce phase. It is also essential to note that Map-Reduce-based PKIP algorithm implementation can outperform C + OpenMP-based PKIP implementation by increasing the number of compute nodes due to the fact that C + OpenMP cannot be implemented over distributed nodes. After performing the initial experiments, we observed that Hadoop block sizes should depend upon the input image dataset. In Hadoop, the data files are divided into blocks. When the block size is too small, the number of data block collaborations involved may increase, thereby affecting the overall results [58].

If we closely observe the execution timing in Tables 10–13, then we can see some similarity between the execution timing of C + OpenMP and single-node implementation of PKIP algorithm for $k = 10$. Moreover, up to 26% reduction could be observed in execution timing for $k = 10$, implemented over 4 nodes if compared to single-node execution timing. However, only up to 4.3% reduction could be observed in execution timing for $k = 4$, implemented over 4 nodes if compared to single node execution timing.

Obtained results illustrate the superiority of the proposed Map-Reduce-based PKIP algorithm when applied to different image datasets. Map-Reduce-based PKIP algorithm enhanced the clustering response time by leveraging on Hadoop framework characteristics. Impact of the number of compute nodes used on the clustering algorithms can also be studied for 3 categories of image dataset by increasing the number of compute nodes (i.e., from 2 to n). Scalability of Map-Reduce based PKIP algorithm can be tested by researchers by maintaining the same type of sample image datasets but varying the cluster size.

With respect to the measure in accuracy of execution timing for PKIP implementation using Map-Reduce, the recorded errors were the best-obtained ones post four runs similar to the case of C + OpenMP implementation. Figure 16 shows the fluctuation or the accuracy in execution time reading for Map-Reduce-based PKIP implementation. For the PKIP implementation using Map-Reduce, the fluctuation in the obtained values were within the range of |1.8%| from the best recorded value. The probable reason for

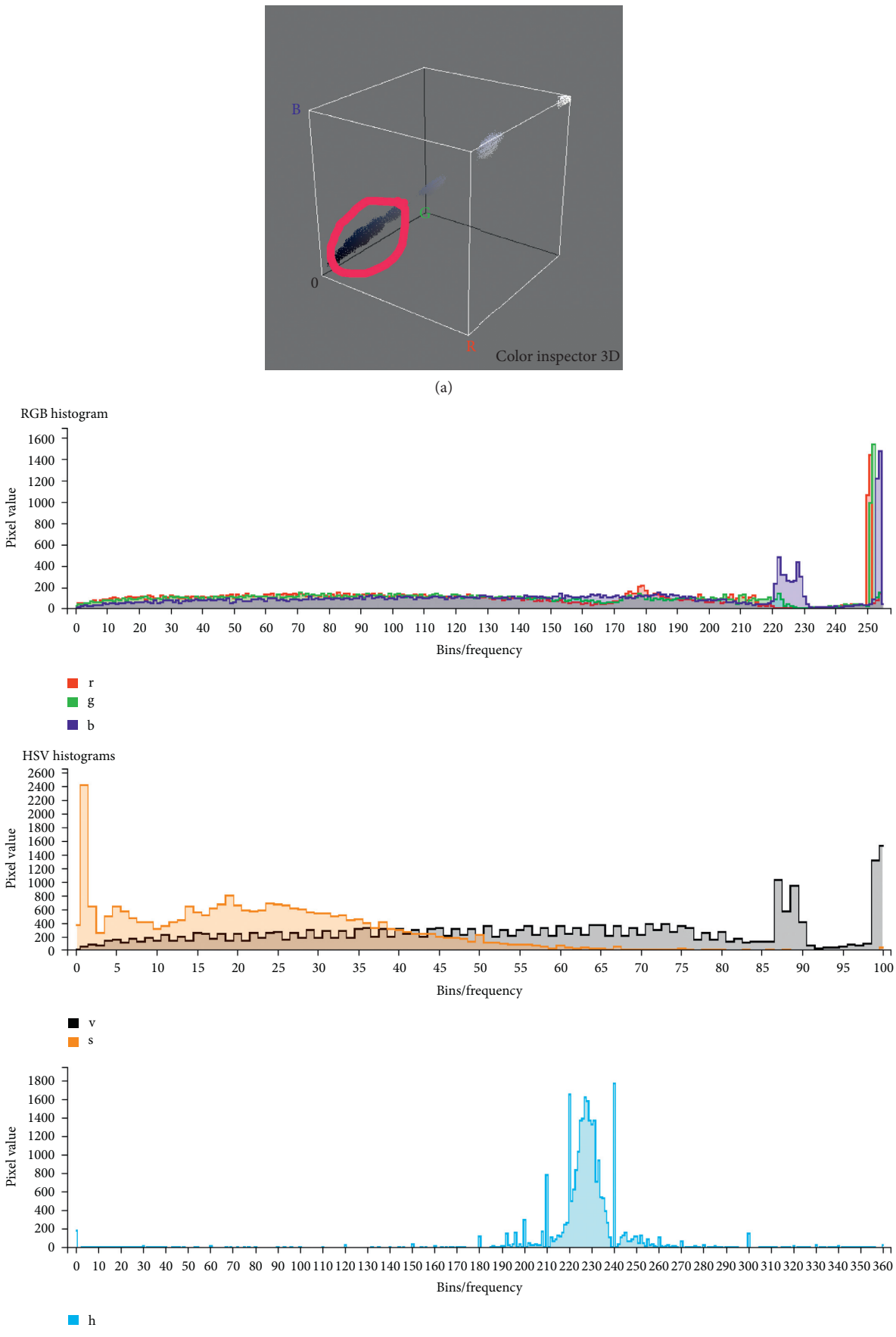


Figure 14: (a) RGB color model; (b) histogram for RGB and HSV color space.

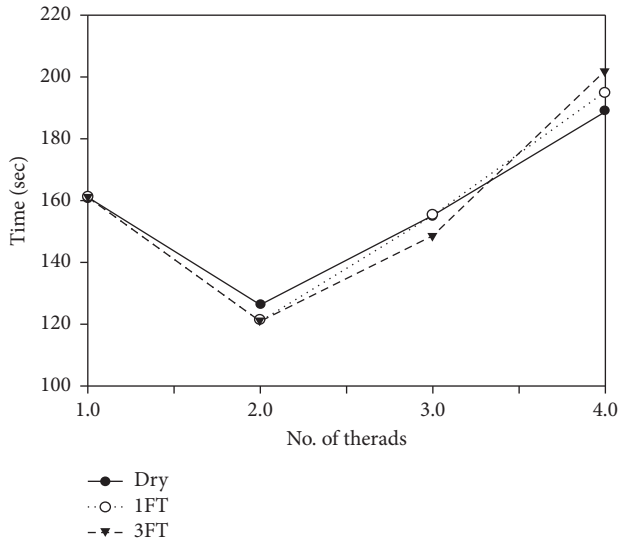


Figure 15: Execution time analysis by varying number of threads.

Table 6: Serial v/s parallel execution time (sec) for dry condition.

Sample	$K = 2$		$K = 4$		$K = 6$		$K = 8$		$K = 10$	
	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec
WP170	29.21	22.53	59.43	46.07	87.65	67.60	120.86	93.14	161.08	127.68
HP180	29.23	22.29	59.46	45.59	87.69	66.89	120.92	92.19	161.15	126.49
HO180	29.22	22.08	59.45	45.16	87.67	66.24	120.90	91.32	161.12	125.40
WO150	29.19	22.36	59.38	45.72	87.57	67.08	120.77	92.44	160.96	126.80
WO160	29.37	22.04	59.75	45.08	88.12	66.12	121.50	91.16	161.88	125.20
WO170	28.18	22.36	57.37	45.72	84.55	67.08	116.74	92.44	155.92	126.81
WP160	29.47	22.04	59.94	45.10	88.42	66.12	121.89	91.16	162.37	125.20
WP150	29.24	22.62	59.49	46.24	87.74	67.86	120.99	93.48	161.24	128.10

Table 7: Serial v/s parallel execution time (sec) for 1FT condition.

Sample	$K = 2$		$K = 4$		$K = 6$		$K = 8$		$K = 10$	
	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec
WP170	29.12	19.69	59.25	40.38	87.57	59.07	120.10	81.76	160.62	113.45
HP180	29.36	22.53	59.73	46.07	88.09	63.60	124.46	90.14	161.82	127.67
HO180	29.32	20.39	59.64	41.79	87.96	61.19	122.29	84.58	161.61	116.98
WO150	29.19	21.06	59.39	43.13	87.58	63.20	120.78	87.27	160.98	120.33
WO160	29.26	22.17	59.52	45.35	87.78	66.52	121.04	91.70	161.30	125.88
WO170	29.45	22.32	59.91	45.65	88.37	66.98	121.83	92.31	162.29	126.64
WP160	29.85	22.46	60.70	45.92	89.55	67.38	123.40	92.85	164.26	127.31
WP150	29.23	19.74	59.47	40.48	87.71	59.22	120.95	81.96	161.18	113.71

Table 8: Serial v/s parallel execution time (sec) for 3FT condition.

Sample	$K = 2$		$K = 4$		$K = 6$		$K = 8$		$K = 10$	
	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec	Serial exec	Parallel exec
WP170	26.00	19.25	53.01	39.51	78.02	57.76	120.03	80.02	145.04	111.27
HP180	29.14	22.04	59.28	45.09	87.42	66.14	120.57	91.19	160.71	125.23
HO180	29.45	22.01	59.91	45.02	88.37	66.03	121.83	91.05	162.28	125.06
WO150	29.44	20.40	59.88	41.81	88.32	61.22	121.77	84.62	162.21	117.03
WO160	29.47	22.19	59.94	45.39	88.41	66.58	121.89	91.78	162.36	125.98
WO170	29.22	22.28	59.45	45.56	87.68	66.84	120.91	92.12	161.13	126.40
WP160	29.37	21.01	59.75	43.02	88.13	63.04	121.51	87.05	161.89	120.06
WP150	29.46	20.43	59.92	41.87	88.38	61.30	121.84	84.74	162.30	117.17

Table 9: Computation of speedup ($k = 10$).

Sample	Execution time using C-Sequential (sec)	Execution time using C + OpenMP (sec)	Speedup
Dry dataset			
WP170	161.087	127.680	1.26
HP180	161.153	126.498	1.27
HO180	161.129	125.403	1.28
WO150	160.964	126.808	1.26
WO160	161.883	125.204	1.29
WO170	155.927	126.810	1.22
WP160	162.370	125.200	1.29
WP150	161.240	128.107	1.25
1FT dataset			
WP170	160.629	113.453	1.41
HP180	161.825	127.679	1.26
HO180	161.616	116.986	1.38
WO150	160.983	120.339	1.33
WO160	161.305	125.882	1.28
WO170	162.292	126.643	1.28
WP160	164.261	127.315	1.29
WP150	161.188	113.712	1.41
3FT dataset			
WP170	145.048	111.279	1.30
HP180	160.713	125.238	1.28
HO180	162.289	125.065	1.29
WO150	162.214	117.035	1.38
WO160	162.365	125.981	1.28
WO170	161.139	126.401	1.27
WP160	161.896	120.068	1.34
WP150	162.309	117.178	1.38

Table 10: Single node v/s multiple Hadoop nodes execution time (sec) for dry condition.

Sample	$K = 4$		$K = 10$	
	Single node	4 nodes	Single node	4 nodes
WP170	69.96	67.02	129.59	103.67
HP180	69.23	66.26	128.38	98.85
HO180	68.55	65.59	127.28	99.27
WO150	69.42	66.44	128.70	102.96
WO160	68.43	65.49	127.07	98.81
WO170	69.42	66.51	128.71	98.85
WP160	68.43	65.55	127.07	96.45
WP150	70.23	67.28	130.02	101.01

Table 11: Single node v/s multiple Hadoop nodes execution time (sec) for 1FT condition.

Sample	$K = 4$		$K = 10$	
	Single node	4 nodes	Single node	4 nodes
WP170	61.13	58.56	115.15	86.12
HP180	65.82	63.06	129.58	98.19
HO180	63.33	60.70	118.73	92.13
WO150	65.41	62.66	122.13	95.27
WO160	68.84	65.95	127.76	99.33
WO170	69.32	66.41	128.53	98.55
WP160	69.73	66.80	129.21	103.37
WP150	61.29	58.71	115.41	86.30

this fluctuation is due to the communication cost of mapping and reducing. Moreover, in this regard, researchers are free to explore the possibility of reducing this fluctuation by optimized scaling and tight scheduling of Hadoop parameters. It is also to be observed that the fluctuation in the execution time reading is lower if compared to C + OpenMP readings. This shows the stability of Hadoop's Map Reduce framework when scaling is taken into account.

Table 12: Single node v/s multiple Hadoop nodes execution time (sec) for 3FT condition.

Sample	$K = 4$		$K = 10$	
	Single node	4 nodes	Single node	4 nodes
WP170	59.78	57.27	112.93	83.57
HP180	68.45	65.57	127.10	94.911
HO180	68.34	65.39	126.93	95.21
WO150	63.36	60.64	118.78	89.23
WO160	68.91	65.98	127.86	96.19
WO170	69.17	66.27	128.29	94.93
WP160	65.24	62.50	121.86	90.59
WP150	63.44	60.78	118.92	95.14

Table 13: Computation of speedup ($k = 10$) for multiple Hadoop nodes v/s C-sequential & C + OpenMP.

Sample	Execution time using C-Sequential (sec)	Execution time using C + OpenMP (sec)	Execution time using 4 nodes (sec)	Speedup (C + OpenMP/4 nodes)	Speedup (sequential-C/4 nodes)
Dry dataset					
WP170	161.08	127.68	103.67	1.23	1.55
HP180	161.15	126.49	98.85	1.27	1.63
HO180	161.12	125.4	99.27	1.26	1.62
WO150	160.96	126.8	102.96	1.23	1.56
WO160	161.88	125.2	98.81	1.26	1.63
WO170	155.92	126.81	98.85	1.28	1.57
WP160	162.37	125.2	96.45	1.29	1.68
WP150	161.24	128.1	101.01	1.26	1.59
1FT dataset					
WP170	160.62	113.45	86.12	1.31	1.86
HP180	161.82	127.67	98.19	1.30	1.64
HO180	161.61	116.98	92.13	1.26	1.75
WO150	160.98	120.33	95.27	1.26	1.68
WO160	161.30	125.88	99.33	1.26	1.62
WO170	162.29	126.64	98.55	1.28	1.64
WP160	164.26	127.31	103.37	1.23	1.58
WP150	161.18	113.71	86.30	1.31	1.86
3FT dataset					
WP170	145.04	111.27	83.57	1.33	1.73
HP180	160.71	125.23	94.911	1.31	1.69
HO180	162.28	125.06	95.21	1.31	1.70
WO150	162.21	117.03	89.23	1.31	1.81
WO160	162.36	125.98	96.19	1.30	1.68
WO170	161.13	126.4	94.93	1.33	1.69
WP160	161.89	120.06	90.59	1.32	1.78
WP150	162.30	117.17	95.14	1.23	1.70

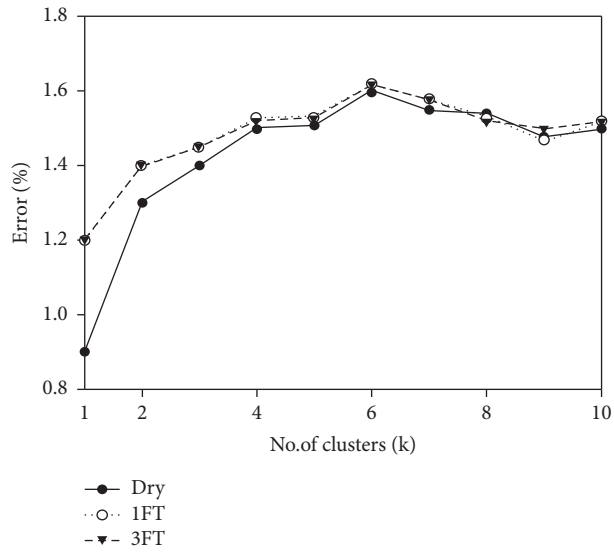


Figure 16: Accuracy of PKIP using map-reduce.

However, in terms of single-node implementation, OpenMP may possess lesser degree of fluctuation if compared to Hadoop subjected to the system specification.

4.3. Precision, Recall, and F-Measure Evaluation. Precision, recall, and F measure are known as the influencing parameters for the evaluation purpose of the proposed PKIP algorithm. The dataset contains 600 image samples comprising of HMA and WMA. The samples were categorized as 300 positive samples and 300 negative samples. The positive samples were the one with adhesion failure, whereas the negative samples were the HMA samples with less than 0.5% adhesion failure. The sample images were cropped into 500×500 pixels and its labelling was done using MS Paint. For the training purpose, the sample images were randomly divided.

Precision is the proportion of genuine positives to the cases that are anticipated as positive. For the proposed experiment, the genuine positives are the pixels that are correctly identified. It is the level of chosen cases that are correct and is computed as follows:

$$\text{Precision} = \frac{\text{correctly identified pixes}}{\text{all detected pixels}}. \tag{14}$$

Recall is the proportion of genuine positives to the all ground truth pixels. The ground truth pixels were the all segmented pixels. Recall is computed as follows:

$$\text{Recall} = \frac{\text{correctly identified pixes}}{\text{all ground truth pixels}}. \tag{15}$$

To determine the weighted average of precision and recall, the parameter of F-measure is used which is computed as follows:

$$F - \text{measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{16}$$

For the initial tests, different patch sizes considered as the container of pixels were used with the dimensions 3×3, 5×5, and 9×9. Table 14 shows the impact of patch size on the influencing parameters. It could be clearly observed that the precision, recall, and F value is almost same for 3×3 and 5×5. For 9×9, the precision values are slightly greater than the other two; however, the execution time is substantially high if compared to 3×3 and 5×5. Therefore, for the training evaluation, we have used 3×3 patch for different values of k. The analysis of influencing parameters has been implemented over single node using sequential processing.

Table 15 shows the impact of k value on the influencing parameter, and it could be clearly observed that $k = 10$ gives the highest F value.

4.4. Analysis of Adhesion Failure Using HSV Thresholding. Tables 16, 17, and 18 represent the percentage of adhesion failure after moisture conditioning (freeze-thaw cycles) and indirect tensile tests for all the k means clustering-based asphalt mixture image samples (i.e., WMA and HMA prepared with PMD and OPC). The images processed using HSV image thresholding for a WMA sample image dataset and HMA sample image dataset for all three moisture sensitivity test is shown in Figures 17 and 18, where the number of clusters has been set to ten ($k = 10$), and the composition of nonzero and zero pixels were computed using equation (13) to determine the percentage of adhesion failure for all asphalt mixture datasets. For the proposed experiment, Scala version 2.11 programming language has been used for the zero and non-zero-pixel computation.

From Tables 16–18 it becomes evident that the percentage of adhesion failure in HMA and WMA sample subjected to different level of moisture conditioning process depends on mixing temperature and also the type of fillers used. The effect of lower mixing temperature on percent adhesion failure of WMA mixtures has consistent increment in adhesion failure at different levels of moisture conditioning. Moreover, for both the categories, i.e., HMA and WMA, the PMD filler comes out to be better if compared with OPC filler, and in this regard, a difference in adhesion failure for up to 4-5% could be observed in 3FT condition. Section 5 discusses some of the key elements regarding the outcome of the results pertaining to PKIP implementation.

5. Discussion

With respect to the analysis of execution timing of PKIP using C + OpenMP and sequentially, it was inferred from Tables 6–8 that the best execution timings are obtained from the processing of WP150, WP160, and WP170 samples. Further research is required to find the reasoning behind the faster execution time for these samples. From one of the recent studies [2], it is essential to note that in terms of stripping analysis using k means algorithm, the larger execution time arises due to classification of dark shaded areas in the image sample. This reason prevails for the proposed experiment also. Moreover, in order to reduce the execution time of the k means clustering, it is not recommended to downsize the resolution of image sample as it results in loss of pixels which will ultimately affect the accuracy of the final results significantly. From the speedup Table 9, it could also be inferred that a maximum speedup of 1.41X is obtained for WP150 and WP170 for the case of 1FT dataset which in turn implies that execution time using C + OpenMP got reduced up to 30% for these datasets.

After analysing the Map-Reduce implementation of PKIP, it could be inferred from speedup Table 13, that for single node implementation, C + OpenMP-based k means implementation is a viable option if k is lesser than or equal to 5 subjected to the specs of the system. However, if we want to increase the number of clusters up to 10+, then Map-Reduce-based implementation of PKIP algorithm is highly suitable as it can give speedup up to 1.86X, i.e., 46.5% reduction in execution timing for the cases of WP150 and WP170 in 1FT dataset category if compared directly with sequential implementation of k means using C Program. Higher gain in speedup is possible if more compute nodes are added to the cluster. For the proposed experiment, only 3

Table 14: Impact of patch size on influencing parameters.

Size of patch	Precision	Recall	F value	Execution time (min)
3×3	0.78	0.84	0.80	34
5×5	0.78	0.85	0.81	46
9×9	0.81	0.87	0.83	87

Table 15: Impact of k value on influencing parameters.

k value	Precision	Recall	F value
2	0.75	0.80	0.77
4	0.76	0.81	0.78
6	0.77	0.81	0.78
8	0.78	0.84	0.80
10	0.79	0.84	0.81

Table 16: Computation for percentage of adhesion failure for dry condition.

Mixture type	Filler	Mixing temperature	Compaction temperature	Sample	Adhesion failure (%)
HMA	PMD	180	170	HP180	2.68
	OPC	180	170	HO180	2.82
WMA	PMD	170	160	WP170	1.51
	OPC	170	160	WO170	1.76
	PMD	160	150	WP160	1.92
	OPC	160	150	WO160	2.10
	PMD	150	140	WP150	2.33
	OPC	150	140	WO150	2.54

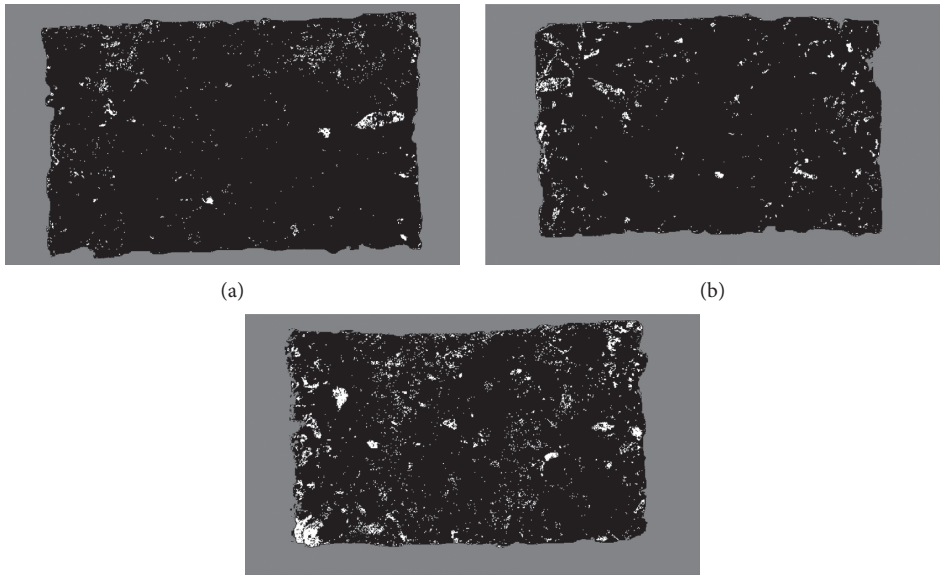
Table 17: Computation for percentage of adhesion failure for 1FT condition.

Mixture type	Filler	Mixing temperature	Compaction temperature	Sample	Adhesion failure (%)
HMA	PMD	180	170	HP180	4.20
	OPC	180	170	HO180	4.52
WMA	PMD	170	160	WP170	3.00
	OPC	170	160	WO170	3.30
	PMD	160	150	WP160	3.43
	OPC	160	150	WO160	3.77
	PMD	150	140	WP150	3.65
	OPC	150	140	WO150	3.98

Table 18: Computation for percentage of adhesion failure for 3FT condition.

Mixture type	Filler	Mixing temperature (°C)	Compaction temperature (°C)	Sample	Adhesion failure (%)
HMA	PMD	180	170	HP180	9.90
	OPC	180	170	HO180	11.00
WMA	PMD	170	160	WP170	5.10
	OPC	170	160	WO170	9.20
	PMD	160	150	WP160	5.50
	OPC	160	150	WO160	9.80
	PMD	150	140	WP150	5.90
	OPC	150	140	WO150	10.10

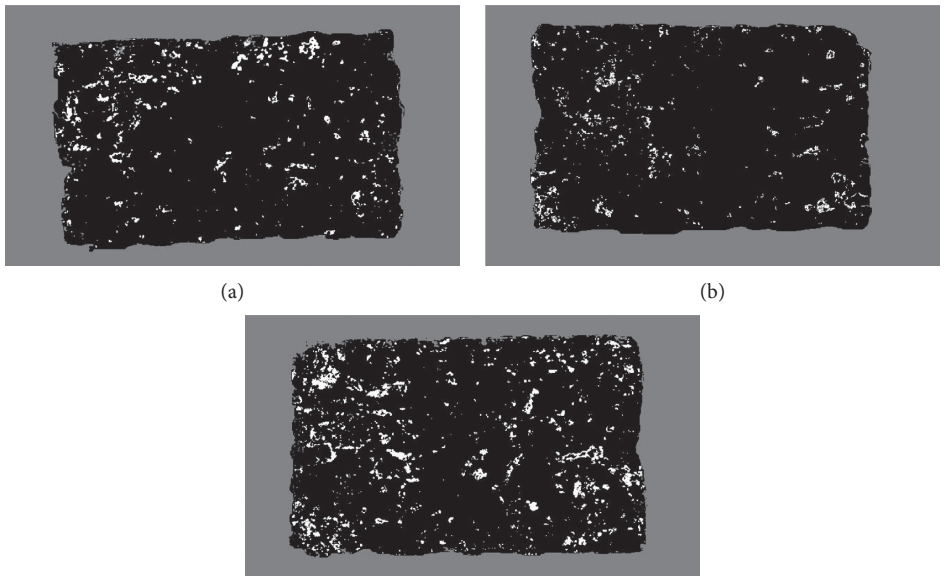
additional compute nodes were added. It is essential to note that Hadoop framework has a vast set of scheduling configuration. Therefore, users can exploit these configurations to exploit the maximum resource benefits from Hadoop and thus explore the possibilities of better execution timing results compared to default scheduling algorithms. This also highlights the fact that more room of optimization prevails in the Map-Reduce algorithms. It is deemed necessary to understand that the execution timing plays a prominent role in clustering process. The proposed Map-Reduce-based PKIP algorithm is well suited for clustering large datasets compared to sequential k means algorithm.



(a) (b)

(c)

Figure 17: Thresholding applied on clustered samples. (a) Dry-WP170, (b) 1FT-WO170, (c) 3FT-WP170.



(a) (b)

(c)

Figure 18: Thresholding applied on clustered samples. (a) Dry-HP180, (b) 1FT-HO180, (c) 3FT-HP180.

It was also be depicted from Tables 16–18 that the WMA specimens prepared using PMD fillers when subjected to 3FT condition show relatively less damage evolution in percentage of adhesion failure if compared to 1FT and dry condition. However, when HMA prepared using PMD filler was subjected to 3FT condition, the evolution of the percentage of adhesion failure was only slightly higher if compared to 1FT and dry conditions. It is to be noted that an environment with indirect light condition should be used in order to prevent light reflections. If such condition is inevitable to avoid, then a heavy usage of advanced filters will be required which will consume higher processing time. A light blue or cream background is suitable to use as a background for the image analysis, as it becomes easier to differentiate between the foreground and the background prior to the implementation of thresholding process. For the proposed experiment apart from differentiating the stripping using adhesion failure, fractured surfaces due to broken aggregate were also observed. Resultant image sample subjected to different freeze-thaw cycles show that large areas of non-zero pixels (brown color pixels) are reduced with the addition of the Cecabase warm mix additive in general. The results therefore indicate that the surfactant-based chemical additive has worked well to substantially reduce the exposure to

moisture in mixes in particular with the use of PMD filler. In addition, the average stripping rate of WMA blend treated with PMD anti-stripping agents was less than 5.5 percent compared to the maximum level of 11% attained by HMA mixture using OPC filler after 3FT cycles. Amongst the different level of moisture conditioning (dry, 1FT, and 3FT) cycles, both HMA and WMA mixtures with OPC filler have shown the highest percentage of adhesive failure (Figure 1(a)). On the other hand, the use of PMD anti stripping agent in both HMA and WMA has shown relatively lower amount of percent adhesion failure in comparison with OPC filler using different conditioning cycles. The analysis of the results indicates that the amount of coating has been increased in mixes that contain PMD filler using WMA additives irrespective of the conditioning cycles used.

6. Conclusion

From the experimental results, it is concluded that the proposed PKIP algorithm is a viable and effective alternative to assess the percentage of adhesion failure in asphalt mixture and thereby encourages an approach to identify the extent of moisture damage in warm mix asphalt. It is important to be noted that in this experiment, high-resolution (5520×3680) images of asphalt mixtures were used to keep the image quality intact. Therefore, when PKIP algorithm was applied on high-resolution asphalt mixture image using OpenMP and Map-Reduce, the execution time got reduced up to 30% and 46% if compared with sequential processing of k means clustering. Nonetheless, researchers are encouraged to apply the proposed PKIP algorithm on quadcore and octacore compute nodes to exploit its efficiency to the best possible extent. Moreover, a distributed version of k means clustering can also be applied in 3D image analysis of asphalt mixture by experimenting with different types of filters [39].

From the results obtained in terms of adhesion failure, with respect to HMA, WMA specimens prepared using PMD filler in dry condition have higher moisture resistance. It is also evident that HMA mixtures with OPC filler after three freeze-thaw (3FT) cycles were more prone to moisture damage. However, the WMA mixture with PMD filler was considered more efficient if compared to 1FT and Dry specimens, as the increase in percent adhesion failure was not too high if compared to Dry and 1FT. The percent adhesion failures in WMA mixtures subjected to 3FT cycles were more prevalent than unconditioned mixtures (Dry) and mixtures subjected to 1FT conditioning, confirming that PG-76 based WMA mixtures when subjected to 3FT conditioning compared to HMA were more efficient. Therefore, to analyse the adhesion failure in asphalt mixtures, PKIP algorithm could potentially be used to quantify moisture damage due to its high accuracy and lower execution time.

Appendix

Hue Saturation Value (HSV) Threshold. The *hue* (H) component of any color represents its similarity to any of the purest form of color, while the saturation (S) component of any color shows the non-white element present in it. The value (V) component of any color shows its subsequent darkness or lightness.

k medians is a variation of k means where instead of calculating mean to compute centroid of the cluster, median is computed.

K-medoid minimizes the sum of dissimilarities between points labelled to be in a cluster and a point designated as the center of the cluster.

Note on Hyperthreading. It is essential to note that the CPU having two physical cores have the capability to process four threads parallelly using hyper threading. However, practically, one physical core is designated to run only single thread but with the help of hyper threading, the CPU is capable to exploit the idle pipeline stages to execute another thread. By doing this, it gives an illusion of a separate physical core to the Operating System. Certain sections of the processor are duplicated to store the architectural state, but the main execution resources are not. A basic RISC processor instruction pipeline comprises of five stages, namely, instruction fetch (IF), instruction decode (ID), memory access (MEM), execute (EX), and register write back (WB). One thread can be in the MEM stage and another thread can be in the IF stage. There are some hazards though. If, for example, the MEM stage is dependent on the EX stage, then it could cause problems. Hyper threading aims at minimizing the number of dependent instructions so that another thread may be run in the idle pipeline stages without dependency issues. With respect to OpenMP code parallelization, data points and cluster centres were marked within the shared construct. Moreover, under the shared construct, the iterative matrix multiplication as mentioned in equation (8) third term was implemented using omp for construct.

OpenMP. Library to implement parallel programming for shared-memory-processors.

Otsu's threshold. An algorithm used for image segmentation which returns a single-intensity threshold which separates foreground and background pixels.

$\mathbf{o}(\mathbf{n}^2 * \mathbf{k} * \mathbf{i})$. Here n is the number of objects, k is the number of cluster and i is the number of iterations.

Scala. A *general* purpose programming language which supports both object oriented and functional paradigm of programming.

Data Availability

The input image samples of asphalt mixtures used in this study are available in the Mendeley data repository via the link: https://data.mendeley.com/datasets/x8ptnsjd7x/1 DOI: 10.17632/x8ptnsjd7x.1. The source code for PKIP is available at https://github.com/nishatakhtar/PKIP.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The proposed experiments have been carried out in School of Aerospace Engineering and School of Civil Engineering of Universiti Sains Malaysia. The authors would like to acknowledge technical staff of Highway Engineering Lab, School of Civil Engineering, Universiti Sains Malaysia, for their kind support in fulfilling this project. The authors would like to acknowledge the RU-Top-Down grant (1001.PAERO.87052) and RUI grant (1001.PAERO.8014035) provided by the Research Creativity and Management Office, Universiti Sains Malaysia, to support this research. In addition, the authors would like to acknowledge Universiti Teknologi Petronas and its Industrial Grant (015MD0-052).

References

[1] Y. Wu and H. Kim, "Digital imaging in assessment of construction project progress," in *Proceedings of the 21st International Symposium on Automation and Robotics in Construction*, IAARC, Jeju, Korea, pp. 537–542, January 2004.

[2] A. Sahari Moghaddam, E. Rezazadeh Azar, Y. Mejias, and H. Bell, "Estimating stripping of asphalt coating using *k*-means clustering and machine learning-based classification," *Journal of Computing in Civil Engineering*, vol. 34, no. 1, Article ID 4019044, 2019.

[3] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[4] N.-D. Hoang, "Image processing-based recognition of wall defects using machine learning approaches and steerable filters," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 7913952, 18 pages, 2018.

[5] M. O. Hamzah, S. Y. Teh, B. Golchin, and J. Voskuilen, "Use of imaging technique and direct tensile test to evaluate moisture damage properties of warm mix asphalt using response surface method," *Construction and Building Materials*, vol. 132, pp. 323–334, 2017.

[6] W. Song, B. Huang, and X. Shu, "Influence of warm-mix asphalt technology and rejuvenator on performance of asphalt mixtures containing 50% reclaimed asphalt pavement," *Journal of Cleaner Production*, vol. 192, pp. 191–198, 2018.

[7] M. R. Kakar, M. O. Hamzah, and J. Valentin, "A review on moisture damages of hot and warm mix asphalt and related investigations," *Journal of Cleaner Production*, vol. 99, pp. 39–58, 2015.

[8] S. Xu, F. Xiao, S. Amirkhanian, and D. Singh, "Moisture characteristics of mixtures with warm mix asphalt technologies—a review," *Construction and Building Materials*, vol. 142, pp. 148–161, 2017.

[9] T. Manzur, K. Mahmood Ehsan, S. Lamia Sultana, and S. Mahmud, "Measurement of surface damage through boundary detection: an approach to assess durability of cementitious composites under tannery wastewater," *Advanced Engineering Materials*, vol. 2016, Article ID 5368635, 13 pages, 2016.

[10] T. S. Yee and M. O. Hamzah, "Evaluation of moisture susceptibility of asphalt-aggregate constituents subjected to direct tensile test using imaging technique," *Construction and Building Materials*, vol. 227, Article ID 116642, 2019.

[11] M. R. Kakar, M. O. Hamzah, and J. Valentin, "Analyzing the stripping potential of warm mix asphalt using imaging technique," *IOP Conference Series: Materials Science and Engineering*, vol. 236, no. 1, p. 012013, 2017.

[12] M. N. Akhtar, J. M. Saleh, and C. Grelck, "Parallel processing of image segmentation data using Hadoop," *International Journal of Integrated Engineering*, vol. 10, no. 1, 2018.

[13] J. Li, S. Song, Y. Zhang, and Z. Zhou, "Robust k-median and k-means clustering algorithms for incomplete data," *Mathematical Problems in Engineering*, vol. 2016, Article ID 4321928, 8 pages, 2016.

[14] H. Song, J.-G. Lee, and W.-S. Han, "PAMAE: parallel *k*-medoids clustering with high accuracy and efficiency," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1087–1096, Halifax, Canada, August 2017.

[15] S. Garg, "Variation of k-mean algorithm: a study for high-dimensional large data set," *Information Technology Journal*, vol. 5, pp. 1132–1135, 2006.

[16] M. O. Hamzah, M. R. Kakar, and M. R. Hainin, "An overview of moisture damage in asphalt mixtures," *Jurnal Teknologi*, vol. 73, no. 4, pp. 125–131, 2015.

[17] L. Li, L. Sun, G. Ning, and S. Tan, "Automatic pavement crack recognition based on BP neural network," *Promet-Traffic & Transportation*, vol. 26, no. 1, pp. 11–22, 2014.

[18] A. Cubero-Fernandez, F. J. Rodriguez-Lozano, R. Villatoro, J. Olivares, and J. M. Palomares, "Efficient pavement crack detection and classification," *EURASIP Journal on Image and Video Processing*, vol. 1, p. 39, 2017.

[19] B. Yu, X. Gu, F. Ni, and L. Gao, "Microstructure characterization of cold in-place recycled asphalt mixtures by x-ray computed tomography," *Construction and Building Materials*, vol. 171, pp. 969–976, 2018.

[20] J. Jiang, F. Ni, Q. Dong, Y. Zhao, and K. Xu, "Fatigue damage model of stone matrix asphalt with polymer modified binder based on tensile strain evolution and residual strength degradation using digital image correlation methods," *Measurement*, vol. 123, pp. 30–38, 2018.

[21] N.-D. Hoang, "Detection of surface crack in building structures using image processing technique with an improved otsu method for image thresholding," *Advances in Civil Engineering*, vol. 2018, Article ID 3924120, 10 pages, 2018.

[22] N.-D. Hoang, Q.-L. Nguyen, and D. Tien Bui, "Image processing-based classification of asphalt pavement cracks using support vector machine optimized by artificial bee colony," *Journal of Computing in Civil Engineering*, vol. 32, no. 5, Article ID 4018037, 2018.

[23] J. Schindelin, C. T. Rueden, M. C. Hiner, and K. W. Eliceiri, "The imagej ecosystem: an open platform for biomedical image analysis," *Molecular Reproduction and Development*, vol. 82, no. 7-8, pp. 518–529, 2015.

[24] S. Bhattacharya, S. Gupta, and K. S. Venkatesh, "Dehazing of color image using stochastic enhancement," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 2251–2255, Phoenix, AZ, USA, September 2016.

[25] Y. Chae, M. Nakazawa, and B. Stenger, "Enhancing product images for click-through rate improvement," in *Proceedings of*

the 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 1428–1432, Athens, Greece, October 2018.

[26] S. A. C. Nelson and S. Khorram, *Image Processing and Data Analysis with ERDAS IMAGINE®*, CRC Press, Boca Raton, FL, USA, 2018.

[27] M. O. Hamzah, M. R. Kakar, S. A. Quadri, and J. Valentin, "Quantification of moisture sensitivity of warm mix asphalt using image analysis technique," *Journal of Cleaner Production*, vol. 68, pp. 200–208, 2014.

[28] S. Dong, J. Zhong, P. Hao et al., "Mining multiple association rules in LTPP database: an analysis of asphalt pavement thermal cracking distress," *Construction and Building Materials*, vol. 191, pp. 837–852, 2018.

[29] X. Zheng, Q. Lei, R. Yao, Y. Gong, and Q. Yin, "Image segmentation based on adaptive k-means algorithm," *EURASIP Journal on Image and Video Processing*, vol. 1, p. 68, 2018.

[30] C. Lantieri, R. Lamperti, A. Simone et al., "Use of image analysis for the evaluation of rolling bottle tests results," *International Journal of Pavement Research and Technology*, vol. 10, no. 1, pp. 45–53, 2017.

[31] Z. Li, G. Liu, D. Zhang, and Y. Xu, "Robust single-object image segmentation based on salient transition region," *Pattern Recognition*, vol. 52, pp. 317–331, 2016.

[32] H. Zhu, F. Meng, J. Cai, and S. Lu, "Beyond pixels: a comprehensive survey from bottom-up to semantic image segmentation and cosegmentation," *Journal of Visual Communication and Image Representation*, vol. 34, pp. 12–27, 2016.

[33] T. Liu, C. K. Liyanaarachchi Lekamalage, G.-B. Huang, and Z. Lin, "An adaptive graph learning method based on dual data representations for clustering," *Pattern Recognition*, vol. 77, pp. 126–139, 2018.

[34] M. Al-Ayyoub, S. Al Zubi, Y. Jararweh, M. A. Shehab, and B. B. Gupta, "Accelerating 3D medical volume segmentation using GPUs," *Multimedia Tools and Applications*, vol. 77, no. 4, pp. 4939–4958, 2018.

[35] M. Capó, A. Pérez, and J. A. Lozano, "An efficient approximation to the k-means clustering for massive data," *Knowledge-Based Systems*, vol. 117, pp. 56–69, 2017.

[36] M. Baydoun, H. Ghaziri, and M. Al-Husseini, "CPU and GPU parallelized kernel k-means," *The Journal of Supercomputing*, vol. 74, no. 8, pp. 3975–3998, 2018.

[37] K. Kerdprasop and N. Kerdprasop, "Parallelization of k-means clustering on multi-core processors," in *Proceedings of the 10th WSEAS International Conference on Applied Computer Science*, pp. 472–477, Budapest, Hungary, December 2010.

[38] L. Bai, X. Cheng, J. Liang, H. Shen, and Y. Guo, "Fast density clustering strategies based on the k-means algorithm," *Pattern Recognition*, vol. 71, pp. 375–386, 2017.

[39] M. N. Akhtar, J. M. Saleh, H. Awais, and E. A. Bakar, "Map-reduce based tipping point scheduler for parallel image processing," *Expert Systems with Applications*, vol. 139, Article ID 112848, 2020.

[40] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image segmentation using k-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015.

[41] T. Zhang and F. Ma, "Improved rough k-means clustering algorithm based on weighted distance measure with gaussian function," *International Journal of Computer Mathematics*, vol. 94, no. 4, pp. 663–675, 2017.

[42] K. Rajeswari, O. Acharya, M. Sharma, M. Kopnar, and K. Karandikar, "Improvement in k-means clustering algorithm using data clustering," in *Proceedings of the 2015 International Conference on Computing Communication Control and Automation*, pp. 367–369, Pune, India, February 2015.

[43] S. Ding, N. Zhang, J. Zhang, X. Xu, and Z. Shi, "Unsupervised extreme learning machine with representational features," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 2, pp. 587–595, 2017.

[44] A. Mohan and S. Poobal, "Crack detection using image processing: a critical review and analysis," *Alexandria Engineering Journal*, vol. 57, no. 2, pp. 787–798, 2018.

[45] M. R. Kakar, M. O. Hamzah, M. N. Akhtar, and J. M. Saleh, "Evaluating the surface free energy and moisture sensitivity of warm mix asphalt binders using dynamic contact angle," *Advances in Civil Engineering*, vol. 2019, Article ID 9153603, 15 pages, 2019.

[46] Public Works Department, Standard Specification for Road Works in Malaysia, Section Four: Flexible Pavements, PWD, Malaysia, 2008.

[47] Asphalt Institute, "Superpave mix design," in Superpave Series, No. 2 (SP-2), 3rd ed., Asphalt Institute, Lexington, KY, USA, 2001.

[48] ASTM, ASTM D1559: Test Method for Resistance of Plastic Flow of Bituminous Mixtures Using Marshall Apparatus. ASTM, West Conshohocken, PA, USA, 2006.

[49] ASTM, ASTM D4867: Standard Test Method for Effect of Moisture on Asphalt Concrete Paving Mixtures. ASTM, West Conshohocken, PA, USA, 2006.

[50] D. Chen, N. Roohi Sefidmazgi, and H. Bahia, "Exploring the feasibility of evaluating asphalt pavement surface macrotexture using image-based texture analysis method," *Road Materials and Pavement Design*, vol. 16, no. 2, pp. 405–420, 2015.

[51] R. Chandel and G. Gupta, "Image filtering algorithms and techniques: a review," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 10, 2013.

[52] E. Ayguadé, "The design of openMP tasks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 3, pp. 404–418, 2008.

[53] T. White, *Hadoop: The Definitive Guide*, O'Reilly Media, Inc., Sebastopol, CL, USA, 2012.

[54] J. Cao, M. Wang, H. Shi, G. Hu, and Y. Tian, "A new approach for large-scale scene image retrieval based on improved parallel -means algorithm in mapreduce environment," *Mathematical Problems in Engineering*, vol. 2016, Article ID 3593975, 17 pages, 2016.

[55] M. Danish, M. N. Akhtar, R. Hashim, J. M. Saleh, and E. A. Bakar, "Analysis using image segmentation for the elemental composition of activated carbon," *MethodsX*, vol. 7, Article ID 100983, 2020.

[56] M. N. Akhtar, J. M. Saleh, and T. Irshad, "Image segmentation using map-reduce framework," in *Proceedings of the 2018 International Conference on Applied Mathematics & Computational Science (ICAMCS. NET)*, Budapest, Hungary, October 2018.

[57] C. Macdonald, R. McCreadie, and I. Ounis, *Agile Information Retrieval Experimentation with Terrier Notebooks*, University of Glasgow, Glasgow, Scotland, 2018.

[58] C. Sreedhar, N. Kasiviswanath, and P. C. Reddy, "Clustering large datasets using k-means modified inter and intra clustering (KM-I2C) in hadoop," *Journal of Big Data*, vol. 4, no. 1, p. 27, 2017.