CHAPTER 4

DESIGN OF PREPROCESSING ALGORITHMS

Preprocessing consists of steps that transform an image into a convenient form that can be readily used by the classifier. It consists of a set of operations that are performed on an image at the lowest level of abstract, which improves the image formation without increasing or losing important image content. Image preprocessing is the operations performed on images at the lowest level of abstraction which doesn't increase image information content. Several types of preprocessing algorithms have been used. They include, but not limited to, resizing, color corrections, noise removal, variation corrections, normalization and ROI extraction or segmentation. The type of preprocessing algorithm used depends on the application under consideration. During the design of ALL-C, two preprocessing algorithms, namely, image enhancement and ROI extraction or WBC identification, are proposed.

4.1. IMAGE ENHANCEMENT

In general, all images contain a certain amount of distortions present, which degrade its visibility or quality (Singh and Verma *et al.* (2021). Generally, the three tasks, denoising, contrast variation correction and edge enhancement are treated as independent tasks, and therefore, three separate algorithms are used. In this research work, all the three works are performed by a unified algorithm. The proposed method is referred to as Unified Contrast adjustment, Edge enhancement and Denoising algorithm or UCED algorithm, in this research work. The algorithms proposed to perform each step are first described followed by the explanation of the method used to combine all three algorithms.

4.1.1. Contrast Enhancement Algorithm

Image contrast is defined as the difference between the intensity levels of two regions in an image (Kundu *et al.* (2023). A microscopic image with low contrast merges light and dark regions, creates a flat or soft image. A low contrast makes it impossible to differentiate two regions, as they are seen as a single region. On the other hand, a microscopic image with high contrast results with dark shadows and bright highlights and fine details of the images might be hidden with bright highlights. The problem of microscopic image degradation because of biased contrast needs to be solved for effective segmentation and feature extraction.

Adaptive Histogram Equation (AHE) is a frequently used algorithm that helps to adjust contrast variations in an image. AHE differs from conventional HE is related to the number of histograms computed. AHE computes several histograms, each corresponding to an independent region of the image, and uses these histograms to redistribute lightness values of the microscopic image. Thus, the AHE is suitable for enhancing local contrast and also definitions of edges in each region of the microscopic image (https://en.wikipedia.org/wiki/Adaptive_histogram_equalization). The AHE has the issue of overamplifying the noise in homogeneous regions of the microscopic image. This issue is solved by using an advanced version of AHE called CLAHE (Contrast Limited Adaptive Histogram Equalization) (Wanga *et al.,* 2004), which can prevent this issue by limiting the amplification.

CLAHE uses a high-frequency filtering that blocks the large scale intensity variations (in other words, low-frequency details), thus enhancing contrast and reduce edge shadowing effect in homogenous areas simultaneously. According to Suharyanto *et al.* (2021), CLAHE is the generalization of AHE and conventional HE. CLAHE starts with the partitioning of microscopic image into contextual regions. In the next step, HE is applied to each of these regions. This step evens out the gray value distribution, which in turn, makes hidden significant features of the microscopic image more visible. The whole image is expressed using the full gray spectrum.

Histogram equalization (HE) is a popular global contrast enhancement method (Umbaugh *et al.*, 1998); Ahirwal *et al.,* 2013) and is the representation of number of pixels for each intensity value in the image. It is widely used, as the algorithm is highly efficient and simple. The histogram used in HE is defined as is a graphical representation of an images' tonal distribution. A histogram of a microscopic image is described as below.

Let the input contrast degraded microscopic image be denoted as M(N, G), where N indicates that the image size is N x N and G refers to the gray levels in the range 0 to L-

1, where L = 1 to 256. A histogram H of M can be defined as a discrete function that maps each G to the number of pixels characterized by that gray level (Equation 4.1).

$$H(G_{r_i}) = n_i \qquad (4.1)$$

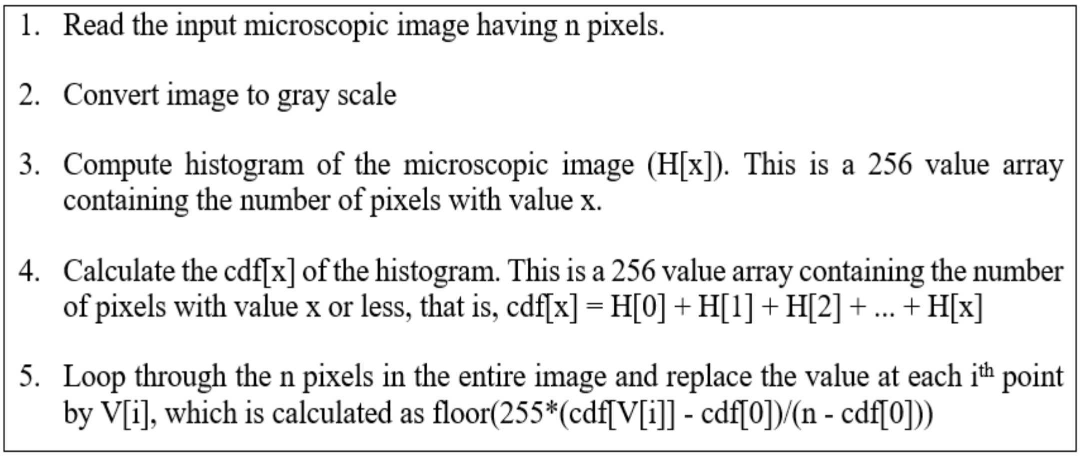
Here, ‘r’ denotes the r^{th} gray level of pixel i and ‘n’ is the number of pixels having values same as G_r.

HE, an approach based on histograms, is a global contrast enhancement technique and results in a histogram that has an approximate constant for all values. HE attempts to create an enhanced version of the degraded image by uniformly distributing the intensity of the pixels over the entire intensity scale. The process also adjusts the contrast of the image by spreading the frequent intensity values again over the intensity scale. In order to equality the brightness in M, first, the pixels of M are normalized to obtain a pdf (probability density function) using Equation (4.2).

$$p_r(G_{r_i}) = n_k/N^2 \qquad (4.2)$$

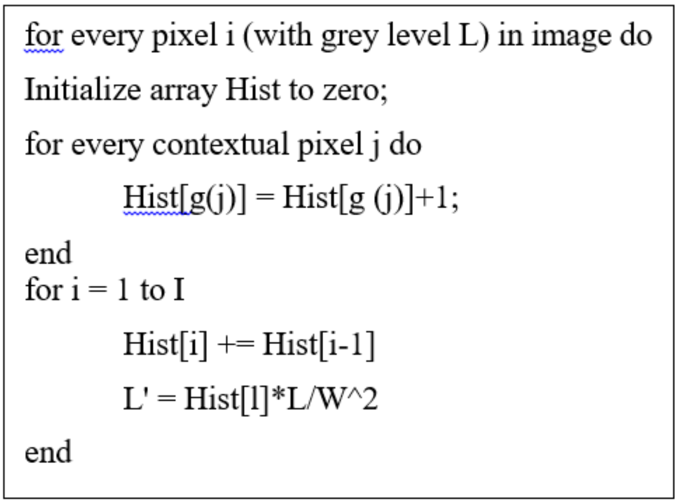
and the above equation satisfies the two criteria $0 \geq p_r(G_{r_i}) \leq 1$ and $\sum p_r(G_{r_i}) = 1$. The conventional HE algorithm is shown in Figure 4.1.

The output of step 5 is the final contrast enhanced microscopic image. The main issue here is that the blocking artifacts introduced, which can be solved through the use of AHE algorithm (Zhiming and Jianhua, 2006) (Figure 4.2).



1. Read the input microscopic image having n pixels.
2. Convert image to gray scale
3. Compute histogram of the microscopic image (H[x]). This is a 256 value array containing the number of pixels with value x.
4. Calculate the cdf[x] of the histogram. This is a 256 value array containing the number of pixels with value x or less, that is, cdf[x] = H[0] + H[1] + H[2] + ... + H[x]
5. Loop through the n pixels in the entire image and replace the value at each i^{th} point by V[i], which is calculated as floor(255*(cdf[V[i]] - cdf[0])/(n - cdf[0]))

Figure 4.1 : HE for Microscopic Contrast Adjustment



```
for every pixel i (with grey level L) in image do
Initialize array Hist to zero;
for every contextual pixel j do
        Hist[g(j)] = Hist[g (j)]+1;
end
for i = 1 to I
        Hist[i] += Hist[i-1]
        L' = Hist[l]*L/W^2
end
```

Figure 4.2 : AHE Algorithm for Microscopic Contrast Adjustment

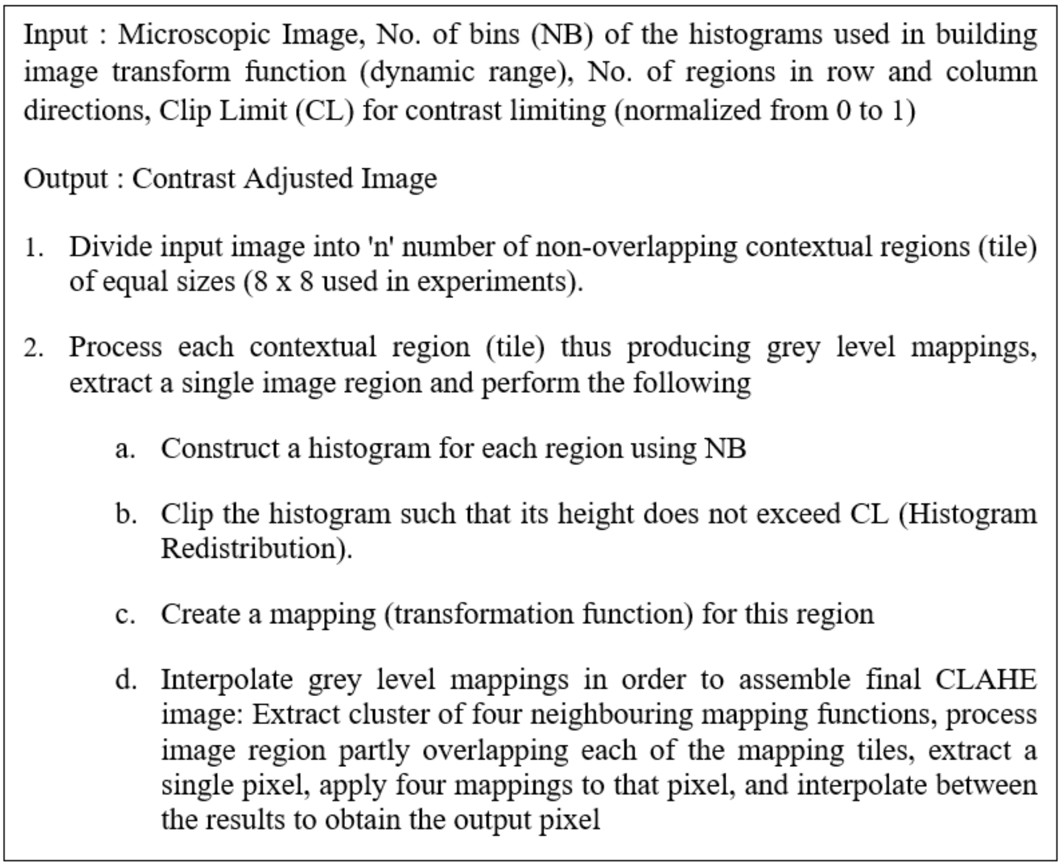
In the AHE algorithm, the number of grey levels is denoted by L, W represents the window size, L and L' are the old and grey level values respectively. Moreover, g(j) represents the intensity of the pixel j. In contrast to HE, the AHE algorithm works on small regions, instead of the whole microscopic image. The contrast of each region is enhanced, so that histogram of the resultant region nearly matches the original histogram. A bilinear interpolation algorithm is then used to combine the neighbouring regions. This step helps to eliminate falsely induced boundaries. The contrast in homogeneous areas can be limited in order to amplify the noise that may be exist in the microscopic image.

The performance of the AHE algorithm depends heavily on the window size. The time complexity of the algorithm is directly proportional the window size. A small window size introduces noise, while a large size introduces artifacts. Selection of correct size is challenging and time consuming. This problem is solved through the use of CLAHE. With CLAHE, the contrast enhancement is defined as the function mapping from input to output intensity. With HE, this the mapping function, m(i), is proportional to the cumulative histogram (Equation 4.3).

$$m(i) = \frac{\text{Display Range} * \text{Cumulative Histogram}(i)}{\text{Region Size}} \qquad (4.3)$$

Here, the derivative of m(i) is proportional to histogram of i. Thus, the focal point of CLAHE is to limit the histogram, which sequentially limits the contrast enhancement.

Figure 4.3 presents the steps involved in CLAHE algorithm (Garg *et al.*, 2011). In the experiments, NB was set to 64, CL was set to 0.01, and tile size used was 8 x 8 and the histogram distribution is Bell-Shaped.



Input : Microscopic Image, No. of bins (NB) of the histograms used in building image transform function (dynamic range), No. of regions in row and column directions, Clip Limit (CL) for contrast limiting (normalized from 0 to 1)

Output : Contrast Adjusted Image

1. Divide input image into 'n' number of non-overlapping contextual regions (tile) of equal sizes (8 x 8 used in experiments).
2. Process each contextual region (tile) thus producing grey level mappings, extract a single image region and perform the following
    a. Construct a histogram for each region using NB
    b. Clip the histogram such that its height does not exceed CL (Histogram Redistribution).
    c. Create a mapping (transformation function) for this region
    d. Interpolate grey level mappings in order to assemble final CLAHE image: Extract cluster of four neighbouring mapping functions, process image region partly overlapping each of the mapping tiles, extract a single pixel, apply four mappings to that pixel, and interpolate between the results to obtain the output pixel

Figure 4.3 : CLAHE Algorithm

4.1.2. Noise Removal

As outlined in Chapter 3, the research work proposes a hybrid algorithm to denoise microscopic images. The hybrid method is designed using two algorithms, namely, DWT and K-SVD, algorithms. This section explains the DWT-based denoising and K-SVD-based denoising methods first, following by the description of the various advantages obtained while combining them along with methodology used to design the hybrid algorithm.

(i) DWT-Based Denoising Algorithm

Discrete wavelet transformation-based denoising algorithm is one of the most popular method used to denoise images in many applications (You *et al.*, 2023). When compared to other transformation-based techniques, the DWT-based algorithms are highly computationally efficient because of its superior localization properties and also offers advantages like sparsity, multi-resolution structure and similarity with the human visual system. The wavelet coefficients are small in magnitude, while the large coefficients coincide with the edges. The DWT-based algorithm removes noise in microscopic images in three main steps (Figure 4.4). They are linear forward DWT, thresholding and inverse DWT.

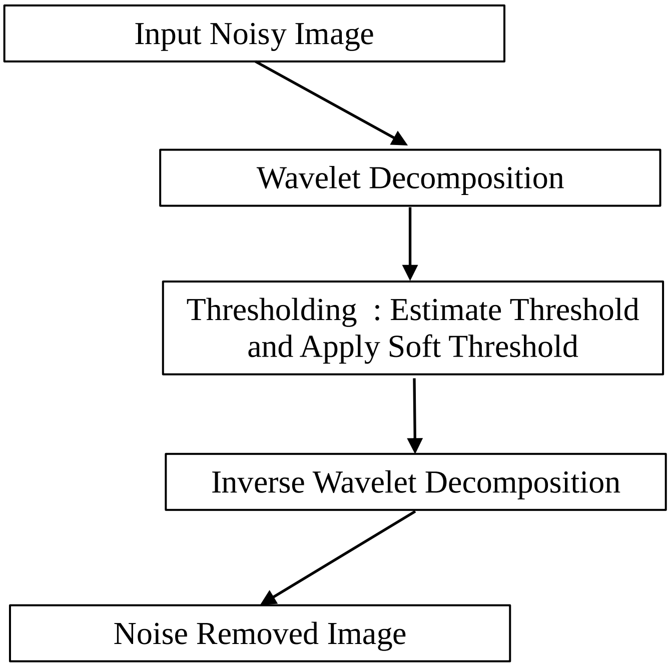


Figure 4.4 : Steps in DWT-Based Denoising Algorithm

a) Discrete Wavelet Transform (DWT)

The most important step of DWT-based denoising algorithm is the selection of forward and inverse wavelet decomposition. Several techniques like Haar, Daubeschies, Coiflets, Symlets, Morlets, Mexican Hat, Meyer and Biorthogonal wavelets are used for this purpose (Ibrahim *et al.*, 2007). This research work uses the Daubeschies (D4) wavelets for denoising. The second point to be considered is the number of decoposition levels. The application of wavelet forward decomposition results in four subbands, namely, LH, HL, LH and HH, obtained from the separable applications of vertical and horizontal coefficients. Here, the LL subband denotes the approximation of the image, while the LH, HL and HH subbands denote detailed coefficients of the microscopic image.

The next level of decomposition can be obtained by further decomposing the LL subband, which results in the 2^{nd} level of decomposition. The level of decomposition is application depended and in this research work, decomposition upto four levels is considered.

b) Thresholding

The second step of DWT-based denoising algorithm is the selection of a wavelet thresholding method (Bruce *et al.*, 1996). Wavelet thresholding is defined as a signal estimation method which analyse the capabilities of wavelet coefficients for denoising. It is mainly used to remove noise present in the microscopic images. Two manners of noise removal can be used with thresholding method. They are kill coefficients or shrink insignificant coefficients in relation to a threshold. The performance of DWT-based denoising algorithm depends heavily on the choice of thresholding method selected. Thus, this step requires two parameters, namely, threshold method (kill or shrink) and selection of a threshold value.

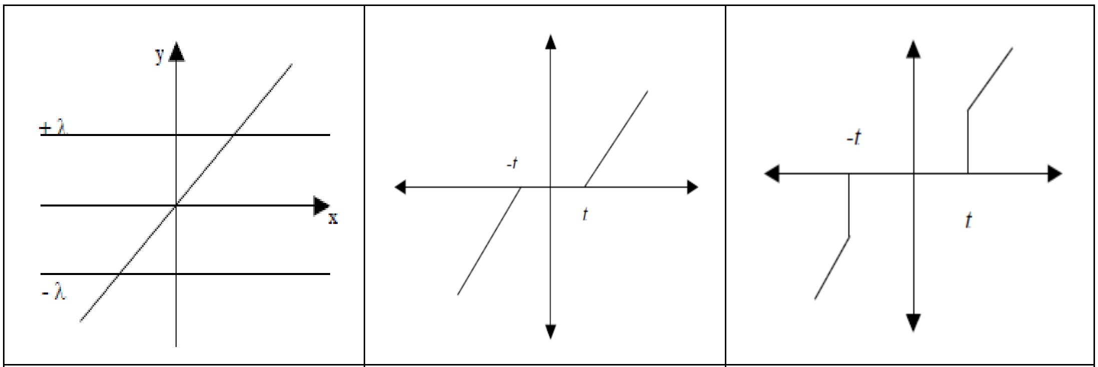
- **Threshold Methods**

Two threshold methods that can be used during denoising are soft and hard thresholding. Equations (4.4) and (4.5) define the working of hard and soft thresholding methods.

$$T_{hard}(I, \lambda) = \begin{cases} I & \text{for all} \mid I \mid > \lambda \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

$$T_{soft}(I, \lambda) = \begin{cases} \text{sign}(I)\max(0, \mid I \mid - \lambda & \text{for all} \mid I \mid > \lambda \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

The hard threholding method use the ‘Kill or Keep’ norm. Here, a pixel of the input microscopic image is retained, it is greater than the pre-defined threshold (λ), else it is set to zero. This method works with the detailed wavelet coefficients only, while keeping the low-resolution wavelet coefficients intact. On the other hand, ‘the shrink or keep’ method is used by the soft thresholding method. Here, if the absolute value of a pixel is less than the threshold, λ, then it is set to zero, else it is replaced by $|I-\lambda|$. Figures 4.5a to 4.5c shows the effect of hard and soft threshold method on an original signal.



(a) Original Signal **(b) Hard Threshold** **(c) Soft Threshold**

Figure 4.5 : Effect of Hard and Soft Thresholding

Discontinuities at ±λ is seen with hard thresholding and they are more sensitive to small changes, while soft threshold avoids both these situations. Thus, the advantages of soft thresholding are, it reduces abrupt sharp changes and provides an image whose quality is not degraded. Because of these advantages, soft thresholding is more frequently used. Once the thresholding operator has been defined, the next step is to address the problem of selecting the corresponding threshold.

- **Threshold Estimation**

The estimation of threshold is the crucial step of DWT-based denoising algorithm and requires careful handling, as a small threshold value produces an image that still has some noisy, while a large threshold value may destroy vital details form the image and may result in a blurred image with edge artifact. To solve these issues, an automatic technique to estimate threshold is much preferred. There are two types of thresholding estimation techniques are available for this purpose. They are, Universal Thresholding also called as global thresholding method, and Subband Adaptive Thresholding, also called as local threhsolding method, exists. The threshold, λ, is estimated using Equation 4.6, when the UT method is used.

$$\lambda = \sigma\sqrt{2\log(\mathrm{M})} \qquad (4.6)$$

where, M denotes the block size of the wavelet domain and σ refers to the local noise variance in each subband of the degraded image after decomposition and is calculated as the average of squares of wavelet coefficients at the highest resolution scale (Equation 4.7).

$$\sigma = \frac{\sum_{i=0}^{N-1} (X_j)^2}{N} \tag{4.7}$$

In adaptive thresholding technique, also called as local thresholding, the threshold is estimated for small regions and therefore will be different for various regions of the image.

There exists several thresholding calculating methods that uses the global and local thresholding technique. For example, VisuShrink method uses global method, while BayesShrink method is most effective and popular local threshold method. This research work uses BayesShrink method.

Bayes Shrink method derives its name from its main principle of reducing the Bayesian risk. The algorithm uses soft threshold and is subband dependent. This means that thresholding is performed at each subband in the DWT decomposition. The method is smothnes adaptive and the threshold, t_B, is derived using Equation (4.8).

$$t_B = \sigma^2/\sigma_s^2 \tag{4.8}$$

In the above equation, the noise variance is denoted by σ^2 and the signal variance without noise is denoted by σ_s^2. The noise variance, σ^2, is determined from HH1 subband using median estimator (Equation 4.9).

$$\sigma^2 = \frac{\mathrm{median}(|g_{j-1,k}|) \text{ where } k = 0,1,\ldots,2^{j-1}-1}{0.6745} \tag{4.9}$$

where $g_{j-1,k}$ corresponds to the detail coefficients in the wavelet transform. As both signals, noise and signal, are independent of each other, it can be described using Equation (4.10).

$$\sigma^2_w = \sigma^2_s + \sigma^2_w \tag{4.10}$$

In the above equation, $\sigma^2 w$ is estimated using the formula in Equation (4.11). Using this variance, σ^2_s is estimated using Equation (4.12).

$$\sigma_W^2 = \frac{1}{n^2} \sum_{x,y=1}^{n} w^2(x, y) \tag{4.11}$$

$$\sigma_S = \sqrt{\max(\sigma_W^2 - \sigma^2, 0)} \tag{4.12}$$

Now, using σ^2 and σ_s^2, the Bayes threshold is estimated using Equation (4.10).

(c) Inverse DWT

The final step of the DWT-based denoising algorithm reconstructs the noise removed wavelet coefficients and is the reverse process of DWT decomposition. During inverse DWT, the enhanced coefficients are first upsampled by placing zeroes in between every coefficient. This effectively doubles its length. They are then convolved with the reconstruction scaling and wavelet filter for approximate coefficients and detailed coefficients respectively. The results are then added together to obtain the final result.

(ii) K-SVD-Based Denoising Algorithm

The K-SVD algorithm to denoise the microscopic images is adopted from the concepts put-forth by Bartuschat *et al.* (2005). K-SVD algorithm can be viewed as a generalization of the K-Means algorithm, which is frequently used for vector quantization. Given a set of N training signals in a matrix $Y \in R^{n \times N}$, the K-SVD method searches for the dictionary D that best represents these signals. It factors the matrix Y into the dictionary $D \in R_{n \times K}$, which contains the K dictionary atoms, and into the sparse coefficient matrix $A \in R^{K \times N}$, which comprises sparse representation coefficients for each of the N signals y (Equation 4.13).

$$\min_{D,A} \sum_i \| a_i \|_0 \text{ subject to } \| DA - Y \|_F^2 \le \varepsilon \tag{4.13}$$

Here, a_i denotes the sparse representation vector corresponding to the i^{th} training signal, ε denotes the error tolerance. The K-SVD algorithm (Aharon *et al.*, 2006) consists of two steps.

Stage 1 : Sparse Coding– it finds sparse representation vectors a_i. Any pursuit algorithm can be used to compute a_i for all training signals.

Stage 2 : Dictionary Update– updates the dictionary such that it best represents the training signals for the found coefficients in A.

Each of the K columns k in D is updated by the following steps of the dictionary update:

- Find the set of training signals that use the current atom, defined by $\omega_k = \{i \mid 1 \le i \le K, a_T^k(i) \neq 0\}$. Here, a_T^k denotes the kth row in A, which corresponds to the atom that is updated.

- Compute the overall sparse representation error matrix E_k. Each of the N columns of this matrix stands for the error of the corresponding training signal, when the kth atom is removed. It is computed using Equation (4.14).

$$E_{k} = Y - \sum_{j \neq k} d_j a_T^j \tag{4.14}$$

- In order to ensure sparsity when computing the representation vectors in the next step, restrict E_k by choosing only those columns that correspond to signals, which use the current atom. That is, choose only columns whose indices are contained in ω_k. The restricted matrix is denoted by E_k^R.

- Apply an SVD decomposition to obtain $E_k^R = U\Sigma V^T$. The first column of U is assigned to the dictionary atom d_k. The coefficient vector is updated to be the first column of V multiplied by the largest singular value $\Sigma(1, 1)$.

The SVD finds the closest rank-1 matrix that approximates E_k. By assigning the elements of this approximation to d_k and a_T^k as shown before, the sparse representation error is minimized.

By performing several iterations of these two steps, where the first step computes the coefficients of the given dictionary and the second step updates the dictionary, the dictionary is adapted to the training signals. The convergence of the K-SVD algorithm is guaranteed, if the sparse coding stage is performed perfectly. In this case, the total representation error $\| DA - Y \|_F^2$, is decreased in each sparse coding step. In addition, the mean squared error (MSE) can be reduced in the dictionary update stage, otherwise it does not change there. The sparsity constraint is not violated. Thus, convergence of the K-SVD algorithm to a local minimum is guaranteed, if the pursuit algorithm robustly finds a solution.

The approximate K-SVD algorithm (Figure 4.6, Rubinstein *et al.*, 2008) starts with an initial dictionary, which is improved in the following k iterations 4. In line 5, sparse representation coefficients for each signal are computed by means of the OMP algorithm or any other suitable algorithm for the current dictionary. Then, each of the K dictionary atoms is updated by a numerically cheaper approximation than the one described above using SVD computation. The first improvement in terms of numerical costs is, that it computes the error matrix only for those signals, which use the atom to be updated. As can be seen in line 7, the error matrix needs not to be computed explicitly. The set of indices of those signals, which corresponds to ω_k from the previous section, is for simplicity denoted by I. The update of the dictionary atoms is performed by optimizing $\| E_I \|_F^2 = \| DA_I - Y_I \|_F^2$, over the dictionary atom and the coefficient vector $\{d_k a_T^k\} = \arg\min_{d_k,\alpha_k} \| E_{I,k} - d_k \alpha_k^T \|_F^2$ subject to $\|d_k\| = 1$.

For simplicity, $(a_T^k, I\}^T$ is denoted by αk. The second improvement in terms of numerical costs is that this problem is solved in the approximate K-SVD algorithm by means of one iteration of a numeric power method. The dictionary atom is obtained by $d_k = \dfrac{E_{I,k}\alpha_k}{\| E_{I,k}\alpha_k \|_2}$ (Line 7). After having normalized the dictionary atom with respect to the i^2 norm in line 8., the nonzero coefficients are computed by $\alpha_k = E_{I,k}^T d_k$ (Line 9). These coefficients are needed for the update of the next dictionary atom. As mentioned before, the dictionary update step converges not to a global solution, but a local solution. Thus, the approximate solution is sufficient, even if the power method is truncated after one iteration.

```
Input : Training Set T, Initial Dictionary $D_0$, Sparse Representation Error Tolerance $\varepsilon$, Number of Iterations K.
Output : Dictionary d and Spare Coefficient Matrix A
Initialize : Set $D = D_0$
for n = 1 .. k do
    $\forall_I : A_i = \underset{\|a_i\|_0}{\arg\min}\, a_i$ subject to $\| y_i - Da \|_2^2 \le \varepsilon$
    for k = 1 to K do
        $D = Y_I\alpha_k - DA_I\alpha_k$
        $d = \frac{d}{\| d \|_2}$
        $\alpha = Y_I^T d - (DA_I)^T d$
        $D_k = d$
        $A_I^k = \alpha^T$
    end for
end for
```

Figure 4.6 : K-SVD Algorithm

In order to remove the noise present in the microscopic image, y, and to recover the original noise free image x, that is corrupted with noise, the sparse approximation problem and K-SVD algorithm described above are used. The K-SVD algorithm trains the dictionary on the image y in order to learn structure present in x. Therefore, the image is decomposed into small patches of size $\sqrt{n} x \sqrt{n}$ stored in vectors $y \in R^n$. The patches can be denoised by finding their sparse approximation for a given dictionary and then reconstructing them from the sparse representation. The denoised patch is obtained from $\hat{x} = \hat{D}_a$, after the sparse representation $\hat{a}$ has been using Eqation (4.15).

$$\hat{a} = \underset{a}{\arg\min} \| a \|_0 \text{ subject to } \| D_a - y \|_2^2 \le T \tag{4.15}$$

Here, T is dictated by ε and the standard deviation of noise σ of the patch. This optimization problem can also be reformulated, such that the constraint becomes a penalty (Equation 4.16).

$$\hat{a} = \arg\min_{a} \| D_a - y \|_2^2 + \mu \| a \|_0 \tag{4.16}$$

The denoising K-SVD algorithm first trains the dictionary on patches of the noisy image Y, and then reconstructs the denoised image X depending on the found dictionary. This can be formulated as an optimization problem (Elad and Aharon, 2006), as given in Equation (4.17)

$$\hat{X} = \arg\min_{X,D,A} \{\lambda \| Y - X \|_2^2 + \sum_{p} \mu_p \| \alpha_p \|_0 + \sum_{p} \| Da_p - R_p X \|_2^2 \tag{4.17}$$

Here, the parameter λ is a Lagrange multiplier that controls how close the denoised output image $\hat{X}$ will be to the noisy image, as can be seen from the first term. The parameter μ_p determines the sparsity of the patch p. R_p denotes the matrix that extracts the patch p from the image. These patches are generally overlapping in order to avoid blocking artifacts at borders of patches.

In the K-SVD denoising algorithm, first the initial dictionary is defined, for example with atoms that contain signals of a discrete cosine transform (DCT), or with noisy patches from the image. The output image X is initialized with the noisy image, X = Y. Then, several iterations of the K-SVD algorithm are performed with the following steps:

- Step 1 : In the sparse coding stage, the sparse representation vectors a_p of each patch R_pX are computed. Here, any pursuit algorithm can be used to approximate the solution of $\forall_p \min \| a_p \| \text{such that} \| Da_p - R_p X \|_2^2 \leq (C\sigma^2)$. The error tolerance is chosen to be the noise variance of the image multiplied by a gain factor C.
- Step 2 : The dictionary update stage (as described earlier) is performed on the patches of the noisy image.

When the dictionary has been trained, the output image is computed by solving Equation (4.18) and the closed-form solution of this simple quadratic term is given in Equation (4.19).

$$\hat{X} = \underset{x}{\arg\min}\{\lambda \parallel Y - X \parallel_2^2 + \sum_{-} \parallel Da_p - R_p X \parallel_2^2\} \tag{4.18}$$

$$\hat{X} = \left(\lambda I + \sum_p R_p^T R_p\right)^{-1} \left(\lambda Y + \sum_p R_p^T D\hat{a}_p\right) \tag{4.19}$$

This solution can be computed by averaging the denoised patches, which are obtained from the coefficients a_p and the dictionary D. Additionally, some relaxation is obtained by averaging with the original noisy image, dependent on the parameter λ.

4.1.3. Edge Enhancement Algorithm

The most fundamental step of image analysis and recognition applications is edge enhancement. Edges are the outline of an object in the image. It can also be described as the boundary between the background and the object. It is defined as the transition in the intensity of the image. Edges play an important role while identifying ROI and its identification consists of two tasks, namely, enhancement and localization. When an edge is enhanced accurately, the location of edges or localization can be found more easily and accurately. Only when the edge location is found, the ROI region can be detected accurately. Thus, ROI region detection performance heavily depends on edge enhancement.

The edge enhancement algorithms enhances the local discontinuities that may exist at the boundaries of the objects in the microscopic image. The algorithms sharpens the boundary of an object with respect to its neighbouring background. The proposed edge enhancement algorithm works to detect edge structures, find weak borders in them and then enhance these weak borders. Edge detection is a popular research topic (Junfeng *et al.,* 2022), while edge enhancement has not received much attention.

The proposed method enhances edges in two steps, namely, edge classification and edge enhancement. In the classification step, first a discrete wavelet transformation is performed to obtain the LL, LH, HL and HH subbands. The edge classification is applied only to HL, LH and HH subands. Now the classification step, classifies the edges into three categories, namely, strong, weak and reverberation artifact. The second step is then applied only on edges classified as weak.

The classification step starts with a feature extraction step, that extracts three texture features, namely, mean (Equation 4.21), variance (Equation 4.22) and correlation (Equation 4.22) using the Gray Level Co-occurrence Matrix (GLCM) procedure (Haralick *et al.* 1973). In these equations, P is the pixel of interest at location i, j and N denotes the number of grey scale levels.

$$\text{Mean} = \mu_i = \sum_{i,j=0}^{N-1} i(P_{i,j}) \qquad \mu_j = \sum_{i,j=0}^{N-1} j(P_{i,j}) \qquad (4.20)$$

$$\text{Variance} = \sigma_i^2 = \sum_{i,j=0}^{N-1} (i-\mu_i)(P_{i,j}) \qquad \sigma_j^2 = \sum_{i,j=0}^{N-1} (j-\mu_j)(P_{i,j}) \qquad (4.21)$$

$$\text{Correlation} = \sum_{i,j=0}^{N-1} P_{i,j}\left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}}\right] \qquad (4.22)$$

After extracting the three features, denoted by $T = \{t_1, t_2, t_3\}$, the features are quantized. The maximum and minimum of each feature set is estimated., using which the size of quantization interval at left (Q_L) (Equation 4.23) and right (Q_R) (Equation 4.25) of the mean values are obtained with a quantization level as 6.

$$Q_L = \frac{2(\mu - \text{Min})}{6} \qquad Q_R = \frac{2(\text{Max} - \mu)}{6} \qquad (4.23)$$

The quantization procedure then estimates the ratio of quantization level (Equation 4.25).

$$K = Q_L/Q_R \qquad (4.24)$$

Using the above calculated values, the quantization Qi of ti is estimated as follow.

- $R = (t_i - \text{Min})/Q_R$ and
- If $R < ½$, then $Q_i = R$ else $Q_i = ((R-1/2)/K) + (l/2)$ $(l=6)$

These quantized features are then grouped into four ranges, namely, small (quantized values between 0 and Q_L), medium (quantized values between Q_L and Q_R), large (quantized values $>Q_R$) and very small (quantized values < 0).

The next step uses these quantized values classifies the edges into weak, strong and artifacts. For this, the selected subband is divided into 8 x 8 fixed sized blocks. The classification step iterates through each of these blocks. Let B_i be the current block analysed. This block is considered as an artefact if Equation (4.25) holds or as strong edges if Equation (4.26) is satisfied or as weak edges if Equation (4.27) is true. In these equation, M, V and C are the three vectors extracted, that is, mean, variance and correlation. While using the above equations, the edge enhancement algorithm can identify high intensity, large variance and high correlation coefficients as strong edges and all coefficients with significant changes in variance and correlation as weak edges.

B_i is Artifact	if $M(B_{i-1})$ && $V(B_{i-1})$ && $C(B_{i-1})$ is in Large category AND $M(B_i)$ && $V(B_i)$ && $C(B_i)$ is in Medium category	(4.25)
B_i has strong edges	If ($M(B_i)$ AND $C(B_i)$ is in medium category) OR ($V(B_i)$ AND $C(B_i)$ is in medium category)	(4.26)
B_i has weak edges	if C1 OR C2 OR C3 OR C4 is true where C1 = $(V(B_{i-1})-V(B_i))$ OR $(C(B_{i-1})-C(B_i))$ is in medium category C2 = $(V(B_i)-V(B_{i+1}))$ OR $(C(B_i)-C(B_{i+1}))$ is in medium category C3 = ($C(B_i)$ is not Very small) AND ($M(B_{i-1})$ && $V(B_{i-1})$ && $C(B_{i-1})$ are in small category) C4 = ($C(B_i)$ is not Very small) AND ($M(B_{i+1})$ && $V(B_{i+1})$ && $C(B_{i+1})$ are in small category)	(4.27)

Next, the sigmoid function (Equation 4.28) is applied on weak edges in order to enhance them.

$$y(x) = \frac{M}{1+e^{-\left(\frac{x-m-\Delta x}{a}\right)}} + \Delta x \tag{4.28}$$

In the above equation M =255 and m = 128 (for 8 bit image). The edge pixel, x, can be in the range -127Δ ≤ x ≤ +128. The parameter 'a' denotes the speed of the change around the center.

4.1.4. Proposed UCED Algorithm

The two algorithms, DWT and K-SVD-based denoising algorithms, are independently effective during noise removal. The DWT-based denoising algorithm provides a simultaneous spatial and frequency domain information of the image and is capable of deconstructing complex signals into basis signals of finite bandwidth, and then reconstructing them again with very little loss of information (Mohideen *et al.*, 2008). However, this algorithm also introduce blocking and ringing artifacts and may degrade fine details, especially in highly noisy images. The K-SVD-based denoising algorithm has high noise-removing capability while preserving fine image details and sharp edges (Chen *et al.*, 2021). However, this method has high time complexity and can work well with gray-scale images only.

Careful scrutiny revealed that both methods treat noisy and noise-free pixels in the same manner. Another common point is that both algorithms focus only on denoising and do not correct contrast variations. Another point that may affect the ROI-segmentation algorithm is concerned with the edges. While both maintain edge details, they do not attempt to correct degraded edges. In order to solve these issues, the combined UCED algorithm is proposed. The proposed UCED algorithm, takes care of contrast adjustment, edge enhancement and noise removal, in a single framework. The steps involved in the UCED algorithm is presented in Figure 4.7.

The first step of UCED algorithm performs contrast enhancement on input noisy image. On the contrast adjusted image, DWT decomposition is performed. The LH, HL and HH subbands are classified as strong, weak and artifact edges. The edge enhancement algorithm is applied on weak edge regions.

- Step 1 : Input Noisy Image
- Step 2 : Use CLAHE to adjust contrast of the image.
- Step 3a :
    - Decompose contrast enhanced image using DWT
    - Classify edges into three categories, namely, strong, weak and reverberation and use sigmoid function on weak edges to enhance edges
    - Perform the following steps to denoise LL subband
        - Divide non-edge regions into noisy and noise free regions using contrast information
        - Perform K-SVD on Noisy Regions
- Step 4 : Perform IDWT

Figure 4.7 : Steps in UCED Algorithm

The next step is to remove the noise present in the image using K-SVD algorithm. The time complexity of this algorithm is reduced as, it is applied only to the non-edge regions of the image. The time complexity can further be reduced, if K-SVD can be applied to a further smaller region. For this purpose, the non-edge region is further divided into two regions, namely, noisy and noise free regions using the contrast information. This manner of applying noise removal algorithm only on noisy region has been proved to be successful by several researchers (Liang *et al.*, 2010; Roman *et al.*, 2005). However, most of the existing method ignores edge enhancement (Fan *et al.*, 2019). In the proposed method, the edges are separated first and the division is performed only on non-edge regions. This further reduces the region where the K-SVD is applied, thus reducing the time complexity of the algorithm. The procedure used to identify noise and noise-free regions in the non-edge detail is given in Figure 4.8.

In the above steps, the local contrast of a block ‘i’ is estimated using Equation (4.29).

$$LC_i = \frac{1}{n_i}\left| \sum_{(m,n)\in i}\left|\frac{LH_{(m,n)}}{LL_{(m,n)}}\right| + \sum_{(m,n)\in i}\left|\frac{HL_{(m,n)}}{LH_{(m,n)}}\right| + \sum_{(m,n)\in i}\left|\frac{HH_{(m,n)}}{LH_{(m,n)}}\right| \right| \tag{4.29}$$

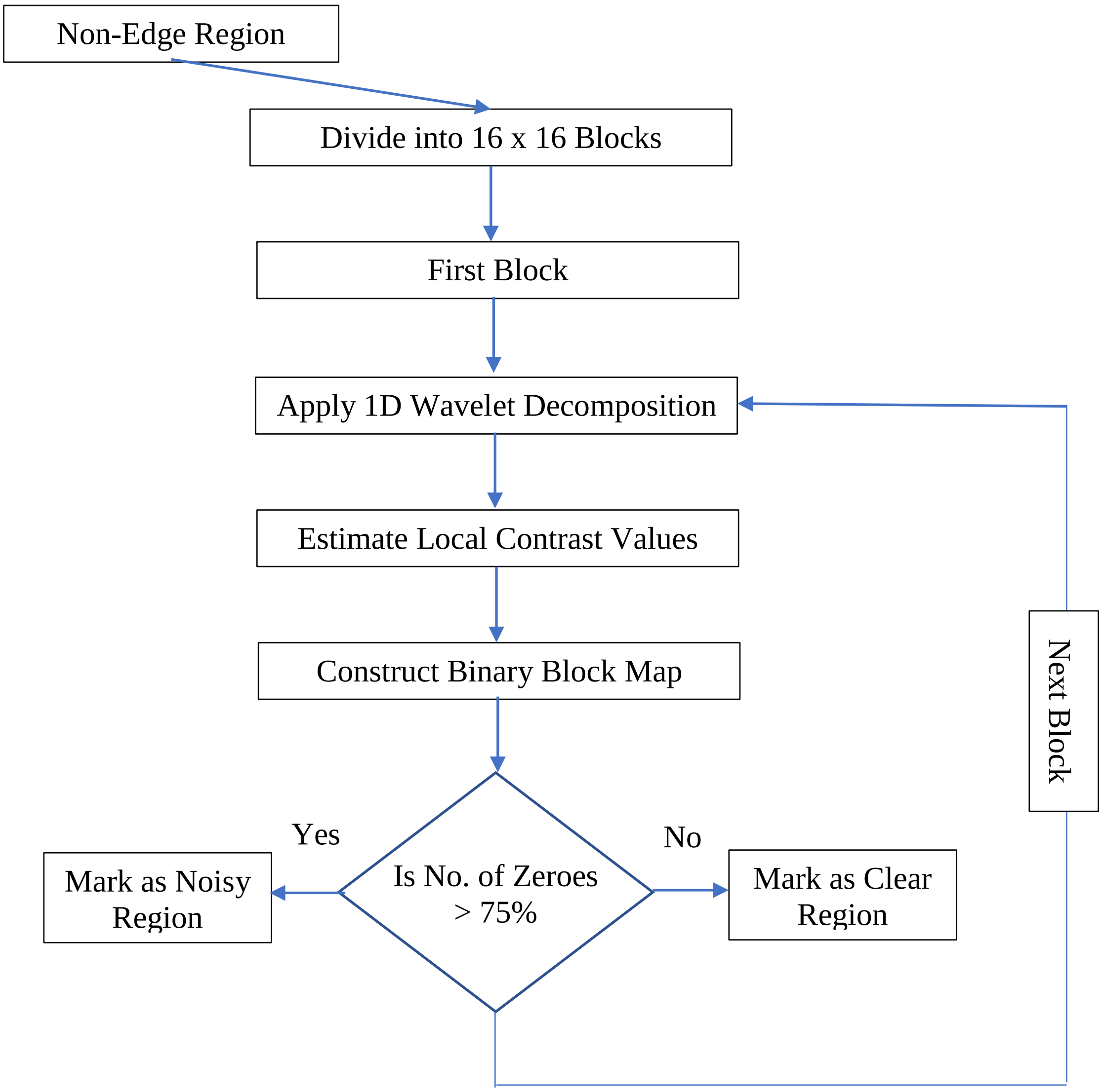


Figure 4.8 : Procedure to Separate Noisy and Noise-Free Regions

In this equation, the nth pixel of block i is denoted as n_i and the size of the suband is mxn. The local contrast of an image block represents its clarity, where a high value denotes a clear block. The binary map based on the local contrast values is constructed using Equation (4.30).

$$\mathrm{BlockMap}(x,y)=\begin{cases}1 & LC_{iA}-LC_{iB}>0\\ 0 & LC_{iA}-LC_{iB}<0\end{cases} \tag{4.30}$$

where LC is the local contrast of the block. The next step applies K-SVD on the regions marked as noisy to obtain noise free or noise reduced regions. The final step of UCED algorithm then uses inverse DWT to obtain the contrast, edge enhanced and noise removed microscopic image.

The proposed algorithm incorporates in a single framework, contrast adjustment, edge enhancement and denoising that exploits the advantages of DWT and K-SVD. The algorithm is efficient in maintaining structure and edge details to the maximum extent. It reduces the execution time considerably, as algorithms focus on specific regions. Further, the proposed algorithm do not introduce artifacts or blur, as noise free regions are not processed at all. All these advantages make it an efficient choice for enhancing microscopic images.

4.2. WBC DETECTION

The most important step of ALL-C system is to quantify the outline of WBC structure in the microscopic image. This is best performed using segmentation algorithms. The usage of segmentation algorithm to retrieve ROI from images is considered mandatory in several medical applications (Mata *et al.*, 2022). The reason behind this is due to the fact the content understanding and disease identification is more accurate and quick when the classification algorithm focus on the ROI region, instead of the whole image. Image segmentation algorithms locate objects and boundaries and assign a label to each pixel, in a way, that all pixels having similar visual characteristics, share a common label. In automatic ALL-C system, the main objective of using segmentation is to identify the WBC (Region-Of-Interest or ROI) in the microscopic image.

The challenges faced are due to the high variations of blood cells in terms of shape, size, edge and position. A microscopic image of blood cells has three colors (Deshpande *et al.,* 2021) as listed below:

- Blue color indicates white blood cells,
- Red color indicates red blood cells, and
- Gray-white color indicates background.

The main objective of the ROI-extraction algorithm is to separate these three cells and it is performed using a segmentation algorithm. A segmentation algorithm is defined as a task that partitions an image into disjoint and homogeneous regions based on some characteristic of the image. During segmentation, care should be taken to create stable segments that are less sensitive to parameter changes.

As mentioned for the reasons mentioned in Chapter 3, Section 3.2.2, this research work applies two enhanced segmentation algorithms to build a combined algorithm that forms final segments that is more stable and accurate. This algorithm is termed as 'Combined Segmentation Algorithm for WBC Identification or CSA_WBC'.

The CSA-WBC is designed using two segmentation algorithms, which are combined synergistically to produce accurate grouping of blood cells. The first algorithm enhances watershed algorithm, while the second enhances a clustering-based algorithm. The steps involved are shown in Figure 4.9.

- Input : Microscopic Image, I
  - Step 1 : Segment I using enhanced watershed algorithm and perform region merging
  - Step 2 : Segment I using enhanced clustering algorithm and perform region merging
  - Step 3 : Combine segment results to produce a single set of segments
  - Step 4 : Use a post processing procedure to further refine the segmented result
- Output : Detected WBCs

Figure 4.9 : Steps in CSA-WBC Identification

4.2.1. Watershed-Based Segmentation Algorithm

The usage of watershed transform algorithm is a popular solution to segment images in the field of morphology

The proposed enhanced watershed-based segmentation algorithm aims to solve the above issues.

(i) Enhanced Watershed Segmentation Algorithm

For a segmentation algorithm to be successful during cell identification and segmentation, it is important to incorporate domain specific features into the segmentation algorithm. Two such important features are color and shape, which combined with the marker algorithm will detect cells more accurately (Arslan *et al.*, 2014). Figure 4.10 shows the various steps involved while enhancing the watershed-based segmentation algorithm.

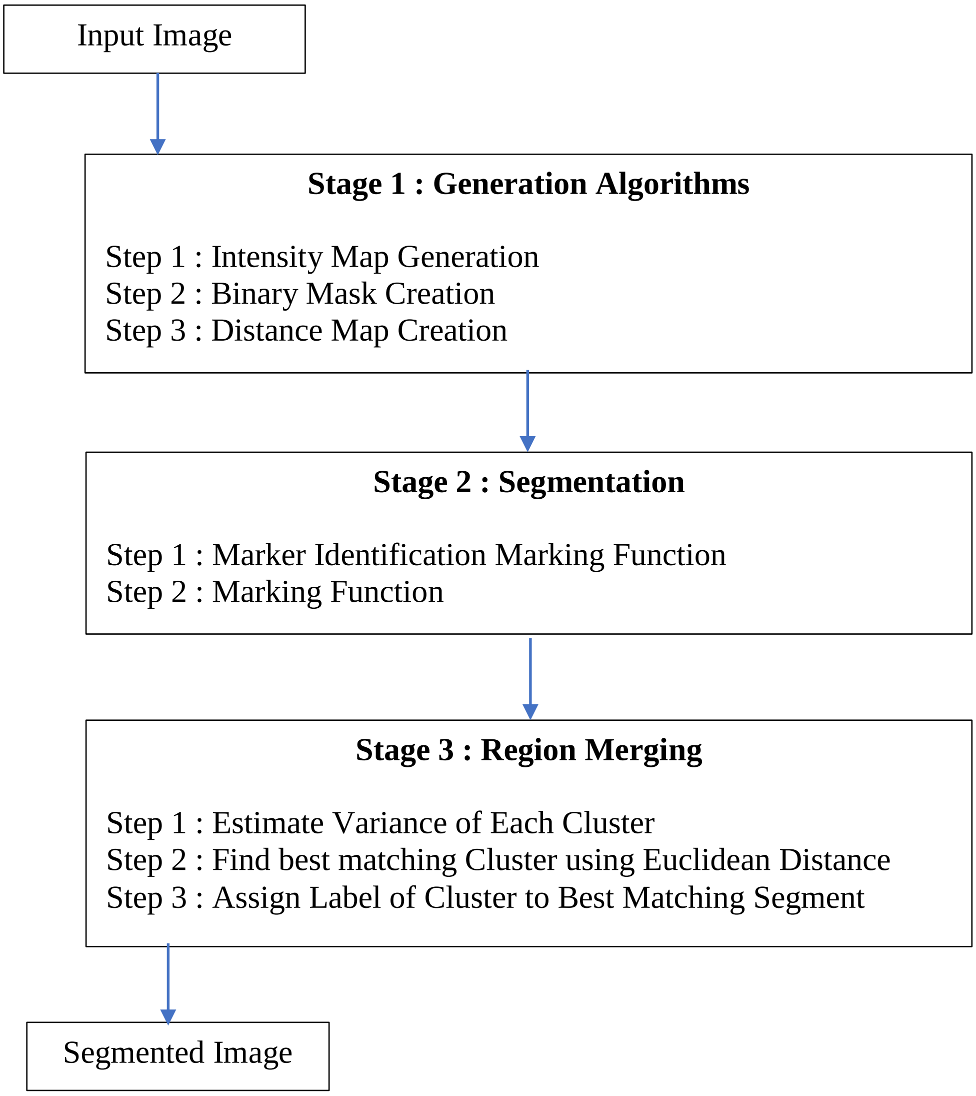


Figure 4.10 : Steps in Watershed-Based Image Algorithm

The enhanced watershed segmentation algorithm begins with a color transformation, followed by the generation of intensity map. From the intensity map, a distance map is constructed, which is used during marker identification. The markers identified are then used by watershed algorithm to create segments. Then, the formulated segments are further fine-tuned to obtain a set of WBCs.

- **Stage 1 : Generation Algorithms**

The generation algorithm can be considered as an initializing algorithm for watershed segmentation. This stage generates the binary mask and distance map required by the watershed algorithm and consists of three steps.

Step 1 : Intensity Map Generation,

Step 2 : Binary Mask Creation, and

Step 3 : Distance Map Creation.

(a) Step 1 : Intensity Map Generation

The first step of the proposed WBC identification algorithm is color transformation. Color transformation is considered as a vital step during segmentation. Most of the existing algorithms use gray scale transformation, which does not always produce quality segments. This is due to the fact that the contrast between ForeGround (FG) and BackGround (BG) is not enough to group similar pixels efficiently. On the other hand, images using RGB color space model store huge amount of details regarding FG and BG characteristics. In a microscopic image, the blue band exhibits a high amount of contrast between FG (cellular regions) and BG. This can be exploited to obtain an optimal threshold level that can distinguish the image pixels. Further, a darker intensity can be envisaged in WBCs in green color band, when compared with other cells. These two features are very much useful during segmentation and hence the first segmentation algorithm using RGB color space. Thus by analyzing the characteristics of the green and blue color bands, an intensity map can be generated using the following steps.

Let M be the input microscopic image defined in the 2D domain $\Omega \to Z^2$. Let M_g and M_b represent the green and blue color bands of M. Further, let $(x, y) \in \Omega$ be the x and y coordinates of a pixel in M. The generator uses Equation (4.31) to generate the intensity map ($M_I(x, y)$).

$$M_I(x,y) = \begin{cases} A & A > 0 \\ 0 & \text{Otherwise} \end{cases} \quad \text{where } A = M_b(x,y) - M_g(x,y) \qquad (4.31)$$

The intensity map helps segmentation in two manners.

1. It makes the WBCs more distinguishable from other objects in IM and BG, and
2. It moves the pixels of other objects and BG more closes, thus helping to distinguish WBCs more efficiently.

The intensity map generated is used by the distance function and marking function of the watershed algorithm.

(b) Step 2 : Binary Mask Generation

The binary mask is generated using Otsu's threshold method (Otsu, 1979) on the intensity map obtained in the previous step.

(c) Step 3 : Distance Transformation

The binary mask obtained in Stage 2 is then transformed into a distance map using Euclidean distance between FG and background pixels. If FG' and BG' are the FG and BG closed sets in the mask and p(x, y) and q(x,y) be any two random points from FG' and BG', then the distance transformation D(p, FG') can be obtained using Equation (4.32).

$$D(p, FG') = \min_{q \in BG'} \{d(p, q), p \in FG'\} \tag{4.32}$$

Here, d(.) is calculated using Euclidean distance between p and q. The distance map thus obtained is used by the markers and marking function of watershed algorithm.

- **Stage 2 : Segmentation**

The enhanced cell segmentation algorithm based on marker-controlled watershed algorithm uses cell boundaries defined by *a priori* information regarding color and shape, obtained from the intensity map and distance map from Stage 1. The three main drawbacks of watershed algorithm are :

(i) Its performance degrades in the presence of noise,
(ii) It often results in over segmentation, and
(iii) Its dependency on the user-defined threshold value.

The usage of denoising algorithm proposed in Task 1 of Phase I solves the first issue and the second issue is solved through the use of distance map generated using intensity map. Finally, the third issue is solved by using an automatic threshold estimation method. Details of the methods used to solve the second and third issues are presented below.

(a) Step 1 : Marker Identification

Marker identification is a vital step in watershed algorithm, as the markers identified correspond to cell locations, from where the flooding starts. When the number of markers identified is more than actual cells, oversegmentation occurs and the opposite will result in under segmentation. Both scenarios are undesirable.

The main goal of using markers is to detect the presence of homogeneous regions from the image by a thresholding technique. They spatially locate object and background, ensures to keep up the interior of the object as a whole. The markers are connected components belonging to an image. Two types of markers are used, namely, internal and external markers. The internal markers are inside each of the objects of interest and external markers are contained within the background.

The marker identification uses h-minima transformation, which has the ability to prevent over segmentation with watershed algorithm. The enhanced watershed algorithm begins by applying h-minima transformation on the inverse inner distance map, in order to reduce undesired minima and reduce false identification of markers. This transform retains only those minima whose depth is greater than a user defined threshold h (set to 2 after empirical evaluations). Next, the remaining regional minima are identified as cell markers. Usage of h-minima before finding regional minima, reduces high number of spurious minima, thus avoids over segmentation.

(b) Step 2 : Marking Function

In the next step, a marking function that represents the topographic surface where water rises is used. This function helps to determine the watershed lines at the locations where two flood meets. In general, color features are used. In the proposed enhanced watershed algorithm, two image features, namely, color and shape of WBCs, are used.

Let R be the regional minima in the inverse distance map D and M_I be the intensity map, defined in $\Omega \rightarrow Z^2$ where $(x, y) \in \Omega$. Initially, the marking function Φ is defined for every pixel, using Equation (4.33).

$$\Phi(x, y) = R(x, y) \,.\, I_M(x, y) \tag{4.33}$$

Then using the marking as initial starting points, this marking function is used during the flooding task of the watershed algorithm to obtain the watershed lines. Segmentation is performed by superimposing the binary mask with these watershed lines.

- **Stage 3 : Region Merging**

To further avoid over segmentation issue, the proposed enhanced watershed algorithm also uses a region merging algorithm to merge similar segments. Region merging apart from avoiding over segmentation also helps to improve segmentation results (Dhanachandra *et al.*, 2019). In this work, the variance of the segments are used to find similar segments. This method was selected because it is simple and effective. A segment is merged to a segment that it best matches, for which distance measure is used. The distance between candidate and other segments is measured using the variance of the segment (Equation 4.34) (Step 1).

$$\sigma_k^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - m_k)^2 \tag{4.34}$$

where n, m_k denotes the total number of pixels and mean of the segment k. Further, σ denotes the variance of the kth segment. The difference between two segments is estimated using Euclidean distance (Equation 4.35) (Step 2).

$$d_{k_i} = \sqrt{|\sigma_k^2 - \sigma_i^2|^2} \tag{4.35}$$

Here, σ_k^2 and σ_i^2 denotes the variance of small and neighbouring segments of segment k. Using the estimated distance measures, the segments with minimum distance are identified and merged (Equation 4.36) (Step 3).

$$d_{best} = \min(d_{ki}) \tag{4.36}$$

On identification of best matching segments, the algorithm changes the label of the matching segment with the label of segment k. This process is repeated for the unprocessed segments.

4.2.2. Clustering-Based Segmentation Algorithm

Another popular and successful way of segmentation is to use clustering algorithms. In this category, several algorithms based on edges, region and density have been proposed. Out of these, region-based algorithms are widely used. In particular, the usage of K-Means clustering algorithm for performing image segmentation is very extensive. K-Means clustering algorithm is a partition-based clustering algorithm, which aims to partition an image into k clusters (or regions) in a manner that places the means of these clusters as far away as possible from each other. It can be considered as a centeroid-based algorithm, where each feature is associated to the nearest mean and belongs to one of the clusters.

In this proposed algorithm, a color-based enhanced K-Means clustering is performed to obtain clusters. Using color features with clustering algorithm is a proven method during the identification of ROI in blood cells (Kazemi *et al.*, 2016). The algorithm begins by converting the microscopic images' color space from RGB model to Lab model. Then, in the lab color space, each pixel of an object has two values, a* and b*, which are used to group pixels into three groups, WBC, RBF and background. The distance between the pixel and each color indicator is estimated using Euclidean distance. Each pixel in the image is labelled to a specific color based on the minimum distance from each color indicator. As the segmentation algorithm is designed to identify the WBCs, the segment labelled as WBC is considered. This is done by identifying a segment that has minimum red color. For this, the average a* and b* values were determined for each cluster centroid and the cluster that has minimum value is determined to have WBC.

The K-Means algorithm (Figure 4.11) is popular because of its easy to implement steps along with producing accurate clusters. However, its efficiency depends upon two input parameters, namely, K and initial seeds or centroids. Here K is the number of clusters to be produced. These two parameters are user-defined and optimal values have to be determined using multiple runs. Moreover, the algorithm has to perform numerous number of computation during similarity calculation using Euclidean distance.

M – Enhanced Input image with resolution (x, y)

p(x,y) – an input pixel to be clustered

- Initialize number of cluster K and centroid c_k.
- For each pixel of an image, calculate the Euclidean distance between the centroid and each pixel of an image using the Equation(4.37).

$$d = \| p(x,y) - c_k \| \tag{4.37}$$

- Assign all the pixels to the nearest centroid based on distanced.
- After all pixels have been assigned, recalculate new position of the centroid using Equation (4.38).

$$c_k = \frac{1}{k} \sum_{y \in c_k} \sum_{x \in c_k} p(x,y) \tag{4.38}$$

- Repeat the process until it satisfies the tolerance or error value.
- Reshape the cluster pixels into image.

Figure 4.11 : Conventional K-Means Clustering Algorithm

Although K-Means has the great advantage of being easy to implement, it has some drawbacks. This research work has identified three most important points that when addressed carefully, can improve the clustering process. They are,

- Mandatory requirement that the number of clusters should be known prior to clustering,
- Sensitive to initial centroid selection, and
- Huge number of computations involved during similarity calculation.

The first point is solved automatically, as the aim of the segmentation algorithm is to group pixels of the microscopic image into three groups, RBC, WBC and background, thus, K is set to 3. The issue of initial centroid selection is solved by using a subtractive clustering algorithm, as a preprocessing step.

Initially, a subtractive clustering algorithm (Banteng *et al.*, 2019) is used to form representative clusters. The subtractive clustering algorithm finds optimal points define cluster centroids based on the density of the surrounding data points. Let X be the set of data points, that is, $X = \{x_1, ..,x_n\}$. The subtractive algorithm treats each point as a cluster center and the potential of data point's x_n is defined using Equation (4.39).

$$P_n = \sum_{i=1}^{n} e^{\frac{-4x_n - x_i^2}{r_a^2}} \tag{4.39}$$

Here, r_a is a constant value greater than 0 representing the cluster radius and is used to define the neighbours. The potential of a data point is measured using Euclidean distance, which indicates that it is a measure of function of distance to all other data points. The r_b should be greater than r_a so as to prevent closely spaced centroids. In general, r_b is always 1.5 times of r_a. Through experimentation, r_a is set to 0.5.

After obtaining potentials, the one with maximum potential is selected as the first centeroid. Let x and p denote the first centroid along with its potential. The potential of the rest of the data points is revised using Equation (4.40).

$$P_n = P_n - P_1 e^{\frac{-4x_n - x_1^2}{r_b^2}} \tag{4.40}$$

In the above equation, r_b is another constant greater than zero indicating the penalty radius, Here, an amount of potential is subtracted from each data point as a function of distance from the first centeroid. This reduces the potential of data points near to the first centroid. This step makes sure that these points are not selected as next centroid. Now the next highest potential is found and is taken as the second centroid. This process is repeated once more, to obtain three cluster centroids. Thus, the enhanced K-Means algorithm works with K = 3 and the initial centroid obtained through subtractive algorithm.

The third issue considered is involved with the computation complexity. The number of computations in K-Means algorithm depends on the number of distance calculations performed to assign each data point to a centroid during each iteration. A computation reduction algorithm is proposed to reduce number of computations, thus reducing time complexity. Let X be a cluster formed in the ith iteration having a data point x. If x is assigned to X after few iterations, then the Euclidean distance from x to each centroid is determined. If the analysis of this distance reveals that it is small from x to X, then the chances of x moving to another cluster is remote. At this juncture, the computation reduction algorithm avoids calculating distance of x from this point. This effectively reduces the number of computations.

The computation reduction algorithm uses one cluster vector and distance vector to store the labels of the clusters and the distance of data points to the nearest centroid in each iteration respectively. The algorithm compares these two vectors during each iteration. If the distance is less than or equal to the old distance, the the data point is connected to the same cluster and further calculations with the rest of the clusters is stopped

Thus, with these optimizations the K-Means algorithm all necessary input parameters are automatically determined without user intervention. Therefore the proposed algorithm is termed as 'Parameterless Fast K-Means Clustering (PFKM) Algorithm'. The steps involved in PFKM is summarized in Figure 4.13. Finally, the region merging algorithm described in the previous section is applied.

4.2.3. Combining Segmentation Results

This research work proposes to combine the results of the two enhanced segmentation algorithms, in order to improve the segmentation accuracy. The results of the two proposed algorithms are combined using a weight majority voting scheme, which is very popular in the classification domain (Molinero *et al.*, 2020; Wardoyo *et al.*, 2020; Dogan and Birant, 2019). The methodology behind using the weighted majority voting scheme for combining segmentation results is described in this section. The steps involved are shown in Figure 4.13.

```
Input : Microscopic Image M  Output : Clusters ($C_1, \ldots, C_k$)
Step 1 :  Assign K =3
Step 2 :  Estimate K initial seeds ($C_j$) using subtractive clustering algorithm
Step 3 :  Repeat
            ▪ For each pixel of an image, calculate the Euclidean distanced,
              between the centroid and each pixel of an image using
              Equation given below.
                          $D = || p(x,y) - c_k||$
            ▪ Find the closest centroid $c_j$ and assign pixels to cluster j
            ▪ Store label of cluster centroid j along with the distance of $x_i$ to
              $c_j$ and store them in array C[] and D[] respectively
            ▪ Set C[i] = j (j is the nearest cluster)
            • Set D[i] = $D_{ij}$ (distance between $x_i$ to closest cluster centroid
              $c_j$)
            ▪ Recalculate Cluster centroids
            ▪ Compute $D_{new}$ of $x_i$ to new cluster centroids.
            ▪ If $D_{new}$ is less than or equal to $D_{ij}$, then
                  $x_i$ belongs to the same cluster j
              else
                  Compute D with every other centroid and assign $x_i$ to the
                  cluster whose D is minimum
                  Set C[i] = j;
                  Set D[i] = $D_{new}$
              end if
          Until convergence
Step 4 :  Output clustered results
```

Figure 4.12 : The PFKM Algorithm

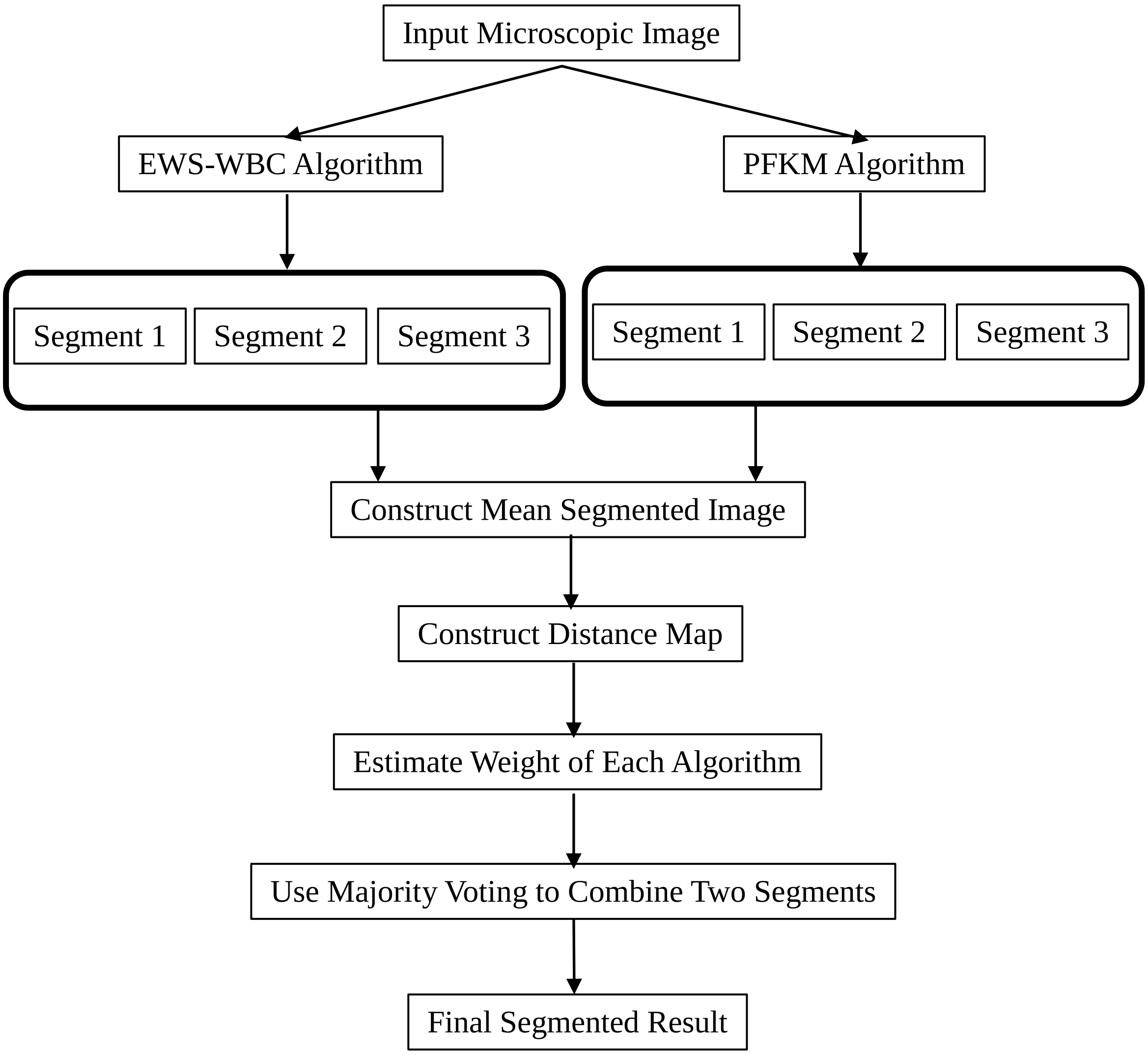


Figure 4.13 : Algorithm to Combine Segments

The input to the proposed combination algorithm are the two segment sets produced by the enhanced algorithms, after applying on the same image. Let the two segment sets produced by enhanced watershed and enhanced K-Means algorithms be denoted as SW and SK respectively. The combination algorithm has three main steps, as described below.

(i) Step 1 : Construct Mean Segmented Image

In general, the segmentation of an image results with an image having the same shape as the original one with a difference. Here, instead of having an intensity value for each pixel position, the segmentation results with an index of the region that the pixel is part of. Different algorithms may produce region indexes that do not match and therefore, while combining, extra processing is required. In this method, instead of having each region's indexed label, a mean intensity value of that region is used (Equation 4.41).

$$R_i(X) = M_{i,j} \text{ for } X \in J \text{ and } M_{ij} = \sum_{X \in J} \frac{I(X)}{|X|} \tag{4.41}$$

This results with a mean segmented image, MS. The image, MS, is very similar to the original unsegmented image, but with segmented details. This steps allows the voting algorithm to work directly on the intensity values, instead of labels.

(ii) Step 2 : Construction of Distance Map, L.

Let MSM denote the mean segmented image of microscopic image, M, obtained while using SM algorithm. Let MSL denote the Mean Labeled Image of M. The comparison is performed using Euclidean distance, on these two results (Equation 4.42).

$$L = \sqrt{\sum_{i,j} (MSL(i,j) - MSM(i,j)^2} \tag{4.42}$$

While using this equation, if the output L has low value, then it indicates a better match.

(iii) Step 3 : Apply Weighted Voting Method

Let LW and LK represent the distance matrix obtained while using Equation (4.43) on watershed and K-Means clustering algorithms respectively. The weights are estimated as the ratio of the inverse proportional to the sum of all L values. As mentioned earlier, the algorithm with small L is best and hence, a high weight has to be assigned to it. The equation for doing this is given in Equation (4.43).

$$W_i = \frac{\frac{1}{L_i}}{\sum_i \frac{1}{L_i}} \tag{4.43}$$

where i = 1 and 2 indicating watershed and K-Means algorithms. Using these weights, a weighted majority voting algorithm is applied to find the final result.

4.2.4. Post Processing

This step uses methods that can further refine the combined segmented results. It consists of two steps, the first is the removal of falsely segmented RBCs and non-WBC blood cells that are similar to WBC. The second method is used to refine the detected WBC regions.

(i) Pruning Red Blood Cells

The segmented results thus obtained have WBCs and partial red blood cells that have similar characteristics. To avoid this, shape features of the segmented regions are used. The usage of shape features helps to identify and remove falsely segmented red blood cells. For this purpose, narrow components are identified and eliminated, which is performed using a circle-fit algorithm (Tosun *et al.*, 2009). Circle fit algorithm locates circles whose radius is higher than a circle threshold, which is set to 20 after empirical evaluations. The proposed algorithm retains only those segmented cells identified by the circle fit algorithm in atleast one circle, and the rest are removed. The next step applies majority filtering using circle shaped kernel with radius W, on the retained cells to smooth their boundaries.

(ii) Refinement of WBCs Detected

In some cases, the segmentation results retrieved only the edges of the blood cell as opposed to the whole image of the blood cell. This issue is solved using a post processing algorithm based on morphological filter. This helps to improve the perceptibility and visibility of the concerned ROI. Care was taken to select a structuring element that was small than the minimum size of the blood cells identified. All cells smaller than this structuring element were removed. The first post processing step is to perform edge enhancement. The edge enhancement is done using the algorithm proposed in Section 4.1.3. The second post processing step uses dilation morphological operator to connect all separated points of the blood cells. This step produces an outline of the perimeter of the blood cells. A 2 x 2 structuring element was used for this purpose. Finally, a hole filling algorithm was used to fill internal holes of the connected elements.

4.3. CHAPTER SUMMARY

In this chapter, the two preprocessing algorithms, proposed to perform microscopic image enhancement and ROI (WBC) segmentation, were described. Using the identified WBC, features are extracted and used to identify ALL using a machine learning classifier. Information regarding the features extracted, method used to construct the optimal feature vector and method to enhance the classification algorithm are presented in detail in Chapter 5, **Classification Using Machine Learning Classifier.**