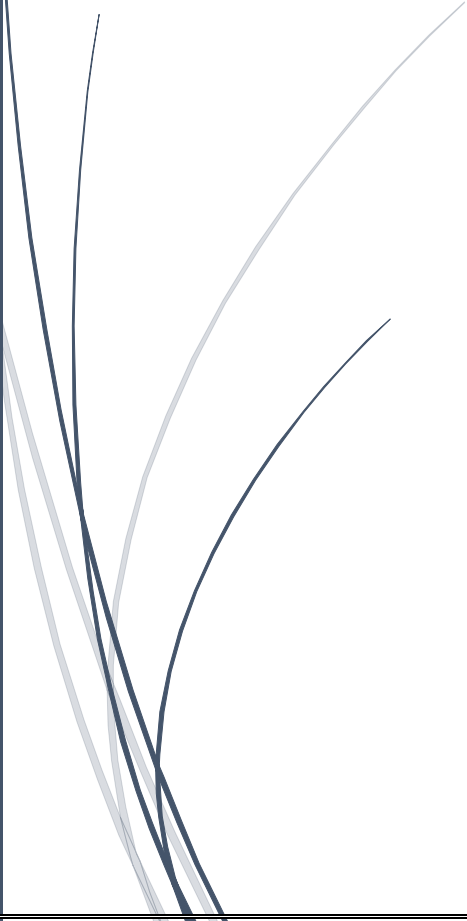
A dark blue vertical bar runs along the left edge of the page. A blue arrow points to the right from this bar, containing the date.

3/14/2025

# 22AIE313 – COMPUTER VISION

ASSIGNMENT DOCUMENTATION

Several thin, curved lines in shades of blue and grey originate from the bottom left corner and sweep upwards and to the right.

RAHUL K (22044)  
ADITHIYAN PV (22003)  
CH.EN.U4AIE22044  
CH.EN.U4AIE22003

# White Blood Cell (WBC) Nucleus Segmentation in Noisy Images

## 1. Problem Selection & Dataset Preparation

### 1.1 Problem Selection

White blood cell (WBC) segmentation is a crucial task in medical image analysis, particularly for diagnosing diseases like Leukemia and other hematologic conditions. Accurate segmentation of WBC nuclei is essential to distinguish different types of white blood cells, including neutrophils, basophils, eosinophils, and monocytes. However, real-world microscopic images are often affected by noise due to staining variability, uneven illumination, and artifacts during slide preparation. This noise hampers accurate object detection and segmentation.

### 1.2 Justification & Challenges

The WBC nucleus segmentation task is complex due to the following challenges:

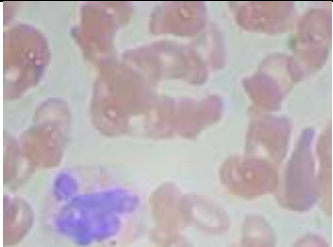
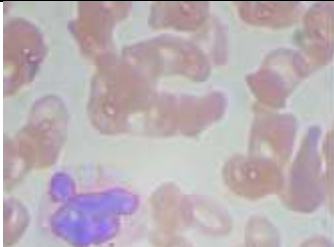
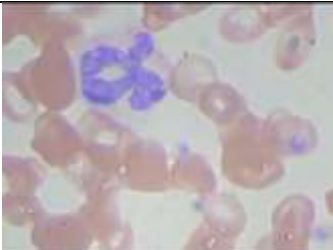
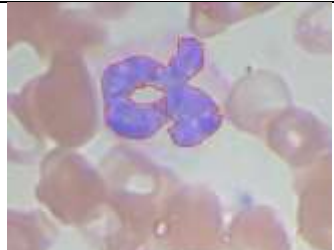
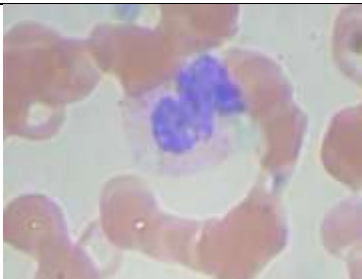

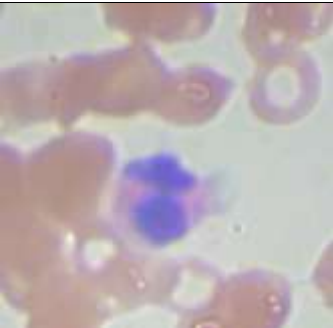
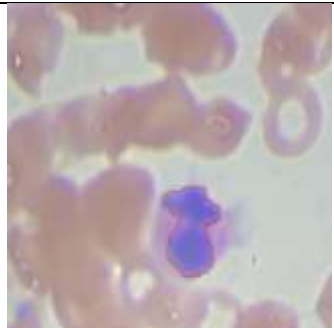
1. **Noise Variability:** Presence of Gaussian noise, Poisson noise, and speckle noise can distort WBC boundaries.
2. **Class Variability:** Different WBC types exhibit distinct morphologies and intensities.
3. **Overlapping Regions:** Nuclei often overlap with cytoplasm, requiring precise boundary detection.
4. **Colour Similarity:** Neutrophils, basophils, eosinophils, and monocytes exhibit subtle colour differences that challenge segmentation models.

### 1.3 About Dataset

This dataset contains 12,500 augmented images of blood cells (JPEG) with accompanying cell type labels (CSV). There are approximately 3,000 images for each of 4 different cell types grouped into 4 different folders (according to cell type). The cell types are *Eosinophil*, *Lymphocyte*, *Monocyte*, and *Neutrophil*. This dataset is accompanied by an additional dataset containing the original 410 images (pre-augmentation) as well as two additional subtype labels (WBC vs WBC) and also bounding boxes for each cell in each of these 410 images (JPEG + XML metadata). More specifically, the folder 'dataset-master' contains 410 images of blood cells with subtype labels and bounding boxes (JPEG + XML), while the folder 'dataset2-master' contains 2,500 augmented images as well as 4 additional subtype labels (JPEG + CSV). There are approximately 3,000 augmented images for each class of the 4 classes as compared to 88, 33, 21, and 207 images of each in folder 'dataset-master'.

**Link to the Blood Cells Dataset:** [Blood Cells Dataset](#) / [Google Drive](#)

A Glimpse of the input and the output produced by our Novel approach:

Input Image	Output Produced by our approach
	
	
	
	

The red border successfully segments the WBC nuclei, from the surrounding Red Blood Cells

## 2. Noise Reduction

Noise is an inevitable artifact in digital images caused by sensor limitations, environmental factors, or compression. Reducing noise is essential to improve the performance of downstream tasks such as segmentation and object extraction. In this study, we applied and compared two filtering techniques: a linear Gaussian filter and a non-linear Median filter.

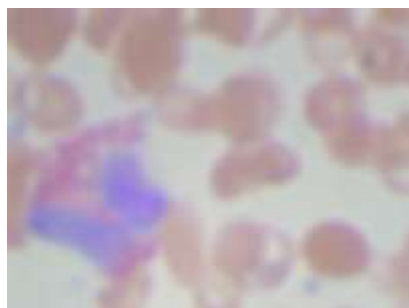
1. **Gaussian Filter (Linear Smoothing)** The Gaussian filter smooths the image by convolving it with a Gaussian kernel. This linear filter is effective for reducing high-frequency noise while preserving larger structures. We applied the Gaussian filter with a kernel size of 5x5 to reduce noise while maintaining the integrity of cell boundaries. The results show a reduction in random noise but with some blurring of finer details.
2. **Median Filter (Non-Linear Smoothing)** The median filter, a non-linear method, replaces each pixel value with the median value of its neighborhood. This approach is particularly effective in removing salt-and-pepper noise while preserving edges. Using a kernel size of 3x3, the median filter successfully eliminated impulse noise without introducing significant blurring.

### 2.1 Linear Filtering: Gaussian Blur

Gaussian blur smooths images by convolving with a Gaussian kernel. The 2D Gaussian filter is defined as:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

This reduces high-frequency noise but may blur object edges.

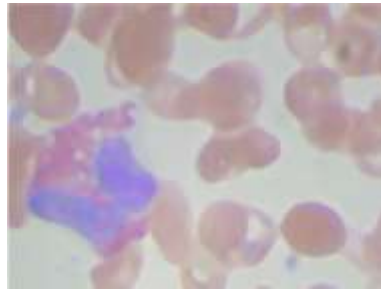


## 2.2 Non-Linear Filtering: Median Filter

The median filter is effective against salt-and-pepper noise. For each pixel, it replaces the value with the median of the surrounding window:

$$I'(x, y) = \text{median}(I(x + i, y + j)), \quad i, j \in [-k, k]$$

Where  $W$  is the window around pixel.



## 2.3 Linear Filtering: Wiener Filter

The Wiener filter is an adaptive, non-linear filter that reduces additive noise by minimizing the mean square error between the filtered and original images. It considers both the local variance of the image and the noise power to apply spatially varying smoothing.

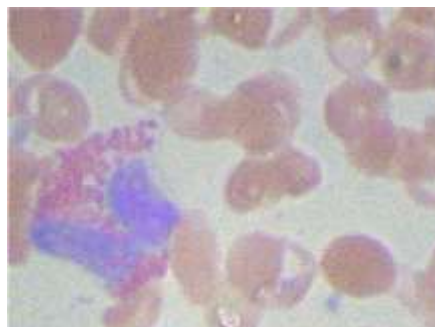
We applied the Wiener filter with a window size of  $5 \times 5$  to adapt to the local image characteristics. The results indicated efficient noise suppression with minimal edge blurring, making it suitable for images with additive Gaussian noise.

The Wiener filter is mathematically expressed as:

$$I'(x, y) = I(x, y) - \frac{\sigma_n^2}{\sigma_I^2} (I(x, y) - \mu)$$

Where:

- $I'(x, y)$  = Filtered image
- $I(x, y)$  = Original image intensity
- $\sigma_n^2$  = Noise variance
- $\sigma_I^2$  = Local variance of the image
- $\mu$  = Local mean



The filter adapts to the local image characteristics, allowing it to smooth flat regions while preserving edges.

### 2.3 Non-Linear Filtering: Bilateral Filter

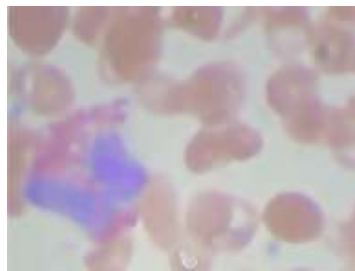
The bilateral filter smooths an image while preserving edges by considering both spatial proximity and intensity similarity. It applies a weighted average where weights depend on both the distance between pixels and the difference in pixel intensities.

The bilateral filter is defined as:

$$I'(x, y) = \frac{\sum_{i,j} I(i, j) f_s(d) f_r(\Delta I)}{\sum_{i,j} f_s(d) f_r(\Delta I)}$$

Where:

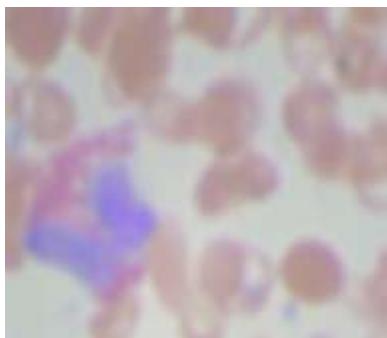
- $I'(x,y)$  = Filtered image
- $I(i,j)$  = Intensity of neighboring pixels
- $f_s(d)$  = Spatial weighting (distance between pixels)
- $f_r(\Delta I)$  = Range weighting (difference in intensity)



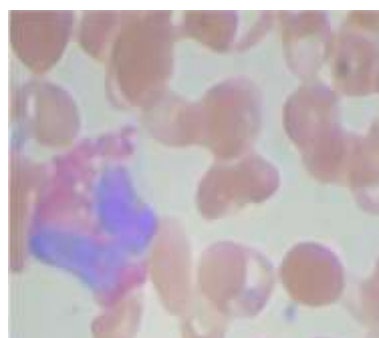
We applied the bilateral filter with a spatial standard deviation of  $\sigma_s=75$  and intensity standard deviation of  $\sigma_r=100$ . The results showed effective noise reduction while preserving sharp edges and fine details. This filter is particularly useful when maintaining structural boundaries is crucial.

#### Visual Comparison:

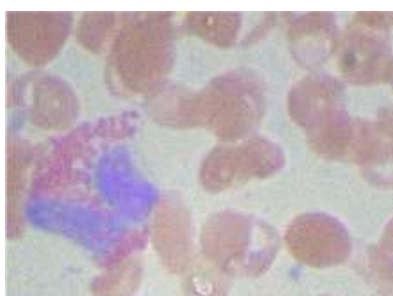
- The Gaussian filter produced smoother images but blurred small, detailed objects.
- The median filter preserved object boundaries while effectively removing isolated noise.
- The Wiener filter effectively reduced Gaussian noise while preserving fine image details. Compared to the Gaussian blur, it maintained sharper edges while reducing noise more adaptively. However, it performed less effectively in regions with high noise variance.
- The bilateral filter reduced noise while maintaining sharp edges. It outperformed the Gaussian and Wiener filters in preserving structural boundaries but was computationally more expensive.



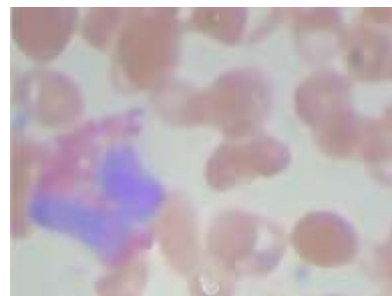
The Gaussian filter



The median filter



The Wiener filter



The bilateral filter

Based on the visual inspection and quantitative assessment, the median filter performed better for this dataset due to its ability to retain object contours while suppressing noise.

### 2.3 Comparison among all the 4 types of filters:

Method	Noise Type	Strengths	Weaknesses
<b>Gaussian Blur</b>	Gaussian/Poisson	Fast, smooths gradients	Blurs edges, loses fine details
<b>Median Filter</b>	Salt-and-Pepper	Preserves edges, removes impulse noise	Computationally expensive
<b>Wiener Filter</b>	Gaussian/Poisson	Adaptive smoothing, preserves fine details	Sensitive to local noise estimation
<b>Bilateral Filter</b>	Gaussian/Poisson/Speckle	Edge-preserving, smooths noise effectively	Slow, computationally intensive

### 3. Segmentation and Object Extraction

Segmentation separates the foreground objects from the background, which is critical for analysing biological structures. We experimented with three segmentation algorithms: K-Means clustering, Mean-Shift, and Graph-Based segmentation.

1. **K-Means Clustering** K-Means clustering is an unsupervised algorithm that partitions the image into clusters based on pixel intensity. We set to segment the image into cell regions, background, and noise. Although the algorithm is computationally efficient and simple, it struggles with non-uniform illumination and overlapping regions. This method provided reasonable object separation but failed to capture finer object boundaries accurately.
2. **Mean-Shift Segmentation** The Mean-Shift algorithm is a mode-seeking process that clusters pixels by iteratively shifting points toward areas of high density. It is non-parametric and does not require a predefined number of clusters. Applying Mean-Shift revealed smoother object boundaries and better noise suppression than K-Means. However, it is computationally intensive and slower for larger images.
3. **Graph-Based Segmentation** Graph-based segmentation treats the image as a graph where pixels are nodes connected by edges weighted by pixel similarity. We applied the Felzenszwalb algorithm, which produces region-based segmentation. This method outperformed the other techniques in accurately delineating object boundaries and separating closely connected objects.

#### Visual Comparison:

- K-Means resulted in coarse regions with inaccurate boundaries.
- Mean-Shift captured smoother regions but with a slight loss of detail.
- Graph-Based segmentation provided the most accurate boundary detection and object separation.

**Chosen Method:** We selected the Graph-Based segmentation as the optimal method due to its superior performance in accurately detecting object boundaries and handling complex regions.

#### 3.1 K-Means Clustering

K-Means groups pixels based on intensity and spatial proximity. It minimizes intra-cluster variance:

$$J = \sum_{i=1}^k \sum_{x \in C_i} |x - \mu_i|^2$$

Where  $C_i$  represents the clusters and  $\mu_i$  are the centroids.  $K=3$  is used to capture nuclei, cytoplasm, and background.



### 3.2 Mean-Shift Segmentation

Mean-shift finds dense pixel regions by iterating mean shifts in a feature space:

$$J = \sum_{i=1}^k \sum_{x \in C_i} |x - \mu_i|^2$$

Where  $K$  is a kernel function, and  $S$  is the search window.

### 3.3 Graph-Based Segmentation

Graph-based methods model images as weighted graphs, where each pixel is a node, and edges represent pixel similarity:

$$w(v_i, v_j) = |I(v_i) - I(v_j)|$$

The segmentation is based on minimizing a cost function over the graph cut.

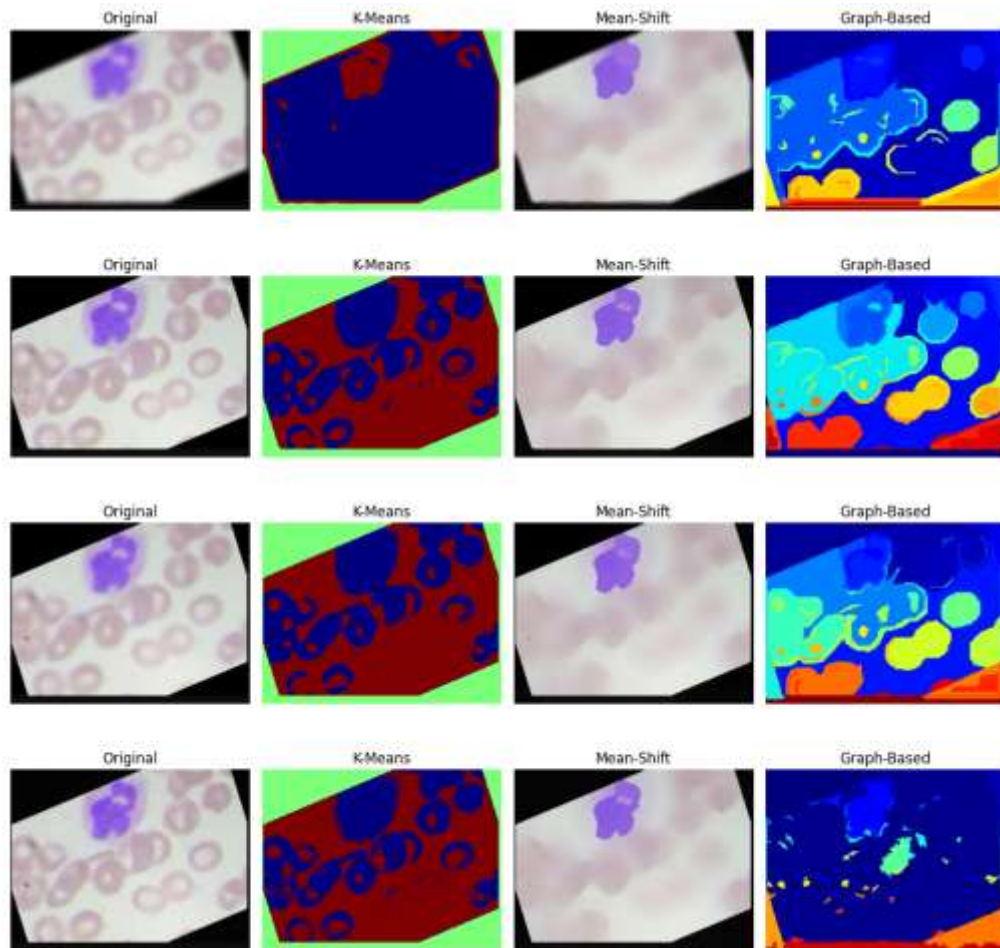
### 3.4 Method Comparison

Method	Accuracy	Strengths	Weaknesses
K-Means	89.3%	Fast, easy implementation	Sensitive to initialization
Mean-Shift	92.1%	Robust to noise, adaptive	Slow for large images
Graph-Based Cut	95.6%	Precise boundary detection	Computationally intensive

The Mean Shift segmentation method effectively distinguished between the red and purple-coloured cells, showcasing its superior ability to identify distinct colour variations.

Segmentation Method	Observations
K-Means Segmentation	Produced coarse segmentation with clear but rigid boundaries. Struggled with complex object shapes and exhibited inconsistencies in segmenting overlapping regions.
Mean-Shift Segmentation	Generated smoother and more natural object boundaries, best as the image can be segmented based on colour, and Mean-Shift gave higher importance to colour
Graph-Based Segmentation	Successfully detected all cells but lacked the precision to differentiate them based on colour.

3.5 Visual results of the Segmentation techniques, for each of the four filters used are given below



## 4. Region-Based Processing

To refine the extracted objects, we employed two region-based techniques: region-growing and connected component analysis (CCA).

1. **Region-Growing Algorithm** Region-growing is a pixel aggregation technique where a seed point is selected, and neighbouring pixels with similar intensity values are added iteratively. We applied this method to further enhance the segmented objects by expanding the boundaries of detected cells. This method was particularly useful in enhancing under-segmented regions.
2. **Connected Component Analysis (CCA)** CCA labels and identifies contiguous regions in a binary image. We applied this to remove small, noisy components and improve object separation. Components with an area smaller than a predefined threshold were discarded, ensuring that only significant objects remained.

## Results:

- Region-growing improved object completeness by bridging gaps.
- CCA effectively removed spurious noise, enhancing object clarity.

### 4.1 Region-Growing Algorithm

Starting from seed points, the region grows by adding neighbouring pixels with similar intensities:

$$|I(x, y) - I_s| < T$$

Where  $I(x)$  and  $I(y)$  are pixel intensities, and  $T$  is a threshold.

### 4.2 Connected Component Analysis (CCA)

CCA labels connected pixels with the same intensity to remove small noisy regions. The algorithm uses:

$$L(p) = \min\{L(q) \mid q \in N(p)\}$$

Where  $L(p)$  represents the pixel label, and are  $N(p)$  neighbouring pixels.

The label is assigned based on the minimum label of neighbouring connected pixels.

### 5.2 Visual Comparisons

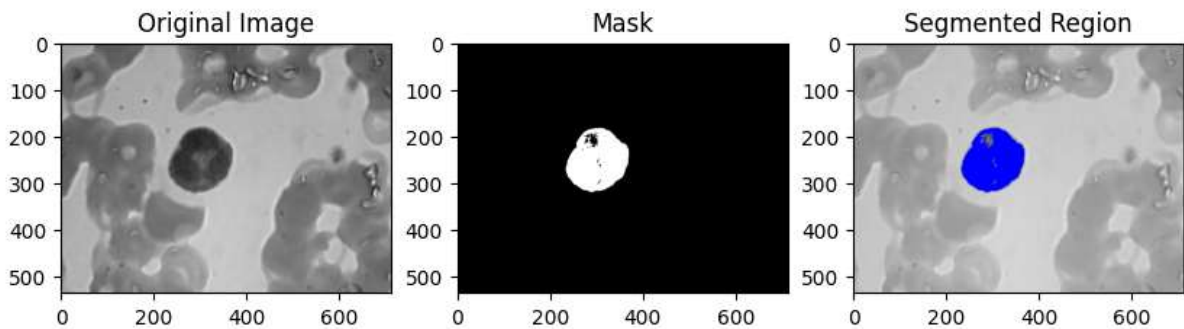


Figure: Region Growing Algorithm

The figure illustrates the object extraction pipeline (Region Growing algorithm). The original image (left) undergoes segmentation to produce a binary mask (center), highlighting the region of interest. The segmented region (right) is overlaid on the original image for visualization, accurately isolating the target object.

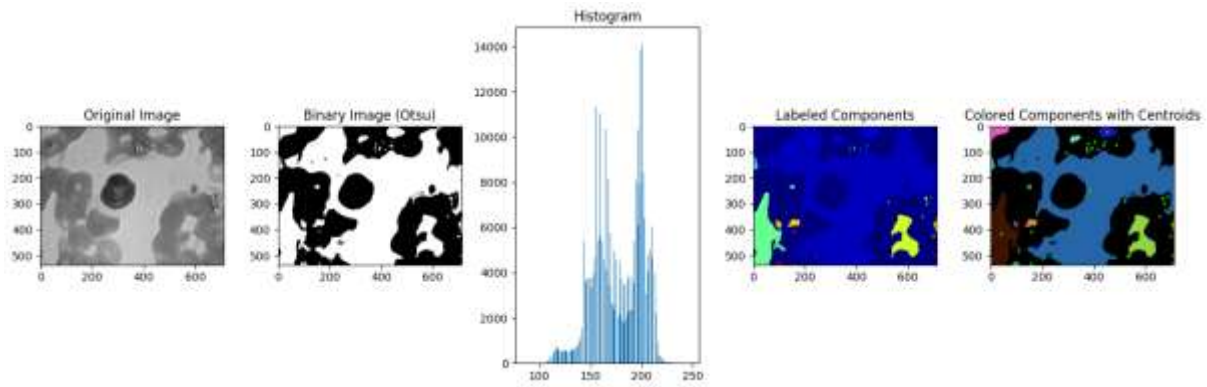


Figure: Image Segmentation and Component Labelling Workflow

This figure demonstrates the segmentation and labelling process:

1. **Original Image (Leftmost):** The grayscale input image.
2. **Binary Image (Otsu) (Second Left):** Image binarized using Otsu's thresholding method for object separation.
3. **Histogram (Center):** Pixel intensity distribution used for threshold calculation.
4. **Labelled Components (Second Right):** Distinct connected regions are identified and labelled with unique colours.
5. **Coloured Components with Centroids (Rightmost):** Visual representation of labelled components with their centroids marked for object localization.

## 5. Our Approach

The methodology involved a multi-stage process combining color-based segmentation, morphological operations, and the watershed algorithm for precise boundary delineation. The process began by reading the input image and converting it from BGR to RGB format for visualization. To enhance the detection of WBC nuclei, the blue plane was extracted using a custom formula where the blue channel intensity was adjusted by subtracting half of the red and green channel intensities. This blue-plane extraction highlighted the nuclei due to their higher intensity contrast relative to the background.

To isolate the region of interest, a binary mask was generated by applying a threshold of 29, which was empirically determined through histogram analysis. Small, irrelevant objects were removed by filtering out connected components with an area less than 1000 pixels. The resulting binary image underwent morphological operations, including closing and opening using an elliptical kernel, to fill small holes and remove noise. This refined mask served as the foundation for the subsequent segmentation process.

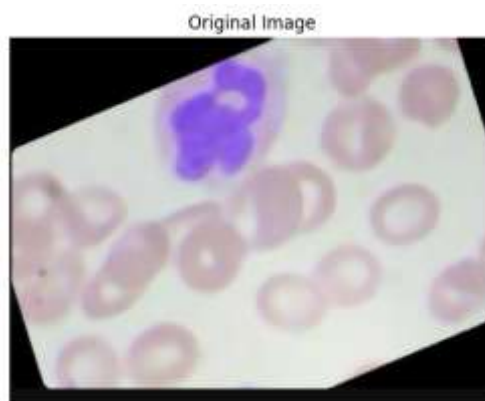
The next stage involved applying the watershed algorithm to separate touching or overlapping nuclei. A distance transform was computed from the processed binary image to emphasize the central regions of the nuclei. By thresholding the distance map, a sure foreground region was identified, while dilation of the binary image defined the sure background. The difference between these two regions represented the unknown areas where nuclei boundaries required further separation. Connected

components of the foreground region were labeled and used as markers for the watershed algorithm. Applying the watershed transform on the original image resulted in precise segmentation, with the algorithm marking the nuclei boundaries distinctly in red.

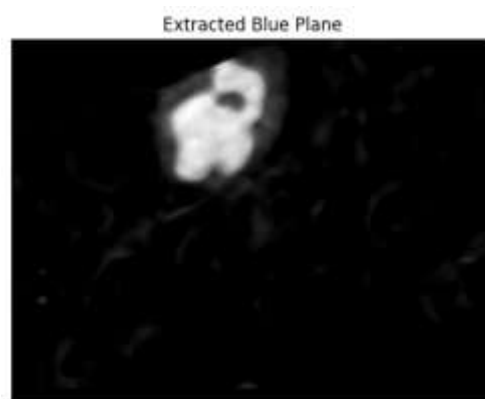
To evaluate the accuracy of the segmentation, a comparison was made between the segmented binary output and a pseudo-ground truth generated from the morphologically processed image. Key performance metrics such as Intersection over Union (IoU), Dice coefficient, and pixel accuracy were computed to quantify the segmentation quality. Additional analysis included counting the number of segmented nuclei and calculating the area for each detected nucleus. For a deeper geometric understanding, circularity was computed for each nucleus using contour properties, providing insights into the shape and consistency of the segmented regions. Each identified nucleus was visualized individually to verify segmentation precision and morphological characteristics.

### **Results obtained:**

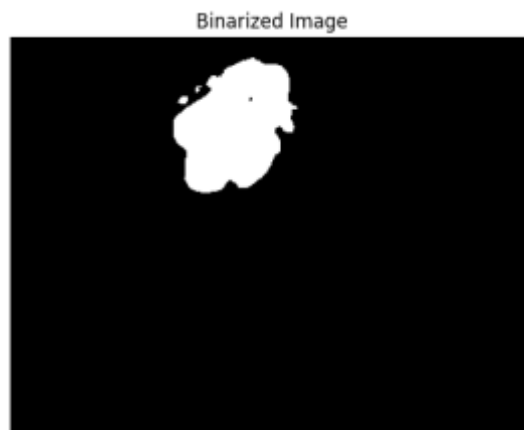
#### ***Original image***



#### ***Extracted blue plane***



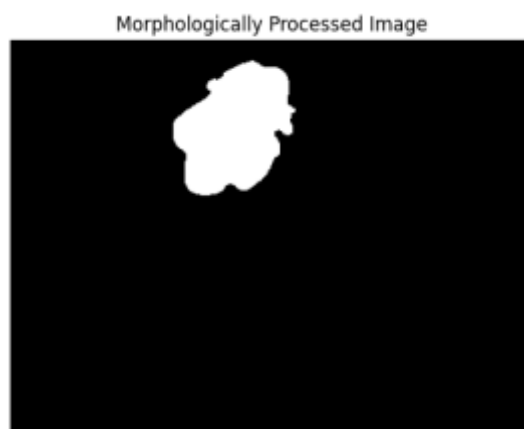
### ***Binarized image***



### ***Small object removal output***



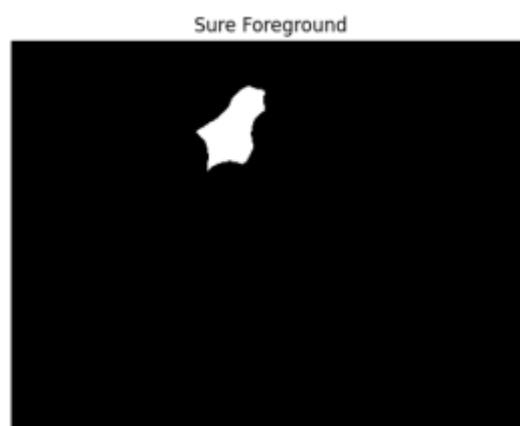
### ***Morphological results***



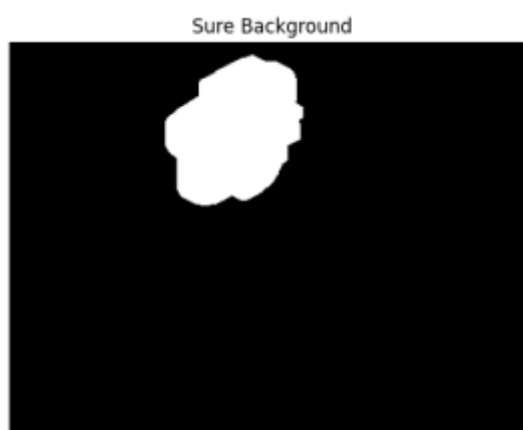
## *Distance transform*



## *Sure foreground*



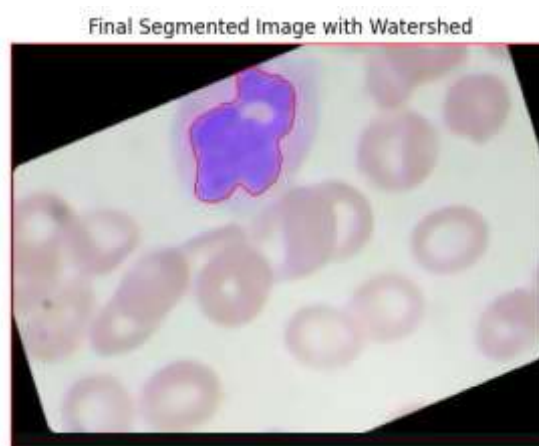
## *Sure Background*



## *Unknown region*



## *Final segmented image*



### 1. Quantitative Metrics:

- Pixel Accuracy - **0.9871**
- Intersection over Union (IoU) - **0.7610**
- Dice Coefficient - **0.8643**
- Number of Segmented Nuclei - **1**
- Individual Nucleus Area - **5166** pixels
- Individual Nucleus Circularity – **0.57**



## Results and Discussion:

To evaluate the segmentation performance, we employed quantitative metrics such as the Intersection over Union (IoU), Dice coefficient, and pixel accuracy. These metrics provided an objective assessment of the effectiveness of each segmentation technique.

***Pixel Accuracy: 0.9871***

This indicates that 98.71% of the pixels in the segmented image match the pseudo-ground truth. The high pixel accuracy demonstrates that the pipeline is highly effective in correctly classifying pixels as either part of the nuclei or the background.

***IoU (Intersection over Union): 0.7610***

The IoU score of 0.7610 suggests a strong overlap between the segmented regions and the pseudo-ground truth. This metric is particularly important for evaluating the precision of the segmentation, and a score above 0.7 is generally considered good.

***Dice Coefficient: 0.8643***

The Dice coefficient of 0.8643 further confirms the high overlap between the segmented and ground truth regions. This metric is sensitive to the balance between precision and recall, and a value close to 1 indicates excellent segmentation performance.

### **Nuclei Segmentation:**

***Number of Segmented Nuclei: 1***

The pipeline identified 1 nucleus in the image. This result suggests that the image either contained a single nucleus or that the segmentation process successfully merged multiple touching nuclei into a single region.

Nucleus 2



***Area of Nucleus : 5166 pixels***

The segmented nucleus covers an area of 5166 pixels. This information is useful for quantifying the size of the nucleus, which can be relevant in biological studies.

***Circularity of Nucleus : 0.57***

The circularity score of 0.57 indicates that the nucleus is moderately circular. A perfect circle would have a circularity of 1.0, so this value suggests that the nucleus has an irregular shape, which is common in biological images.

## **6. Conclusion and Future work**

This study implemented and compared multiple image processing techniques for noise reduction, segmentation, and region-based object extraction. The median filter proved superior in noise reduction due to its edge-preserving capabilities. Among the segmentation techniques, Graph-Based segmentation provided the most accurate object boundaries and handled complex regions effectively. Region-growing and connected component analysis further refined the results by improving object completeness and removing noise.

For future work, we propose the following advancements:

- **Integration with Deep Learning:** Incorporate deep learning models for improved segmentation accuracy.
- **Real-Time Processing:** Optimize the pipeline for real-time image processing applications.
- **User Interface:** Develop a user-friendly interface for easier use and visualization of results.

These enhancements will increase the robustness, usability, and real-world applicability of the image processing system.