Blood Cell Segmentation

Ahmet Ezdesir
Istanbul Technical University
Computer and Informatics Department
Email: ezdesir19@itu.edu.tr

Abstract – The project's goal is to create a blood cell counting and segmentation algorithm that counts the amount of blood cells in each sample using image processing techniques. To detect and count distinct types of blood cells in the sample, the system will go through multiple steps, including image preprocessing, segmentation, feature extraction, and classification. The purpose of this project is to develop a dependable and efficient instrument for healthcare professionals to count blood cells rapidly and accurately, which can assist in the diagnosis and treatment of a variety of medical diseases.

1. Introduction

Counting blood cells is a crucial aspect of medical diagnosis and treatment. Precise cell counting is essential in identifying and diagnosing various medical ailments. On the other hand, traditional manual microscopic examination methods for blood cell counting are time-consuming, labor-intensive, and susceptible to human error. To overcome these limitations, computer-based blood cell counting algorithm that employ image processing have been developed. In recent years, there has been an increased application of image processing and machine learning techniques to enhance the accuracy and efficiency of blood cell counting. The objective of this project is to create a blood cell counting system that employs image processing and machine learning techniques to accurately count the number of blood cells in each sample. The system will undergo several stages, including image preprocessing, segmentation, and classification, to detect and count different blood cell types in the sample.

2. Proposed Approach

The project consists of several stages as stated above. Let's explain them in detail. As we have collected sample blood cell images, it is necessary to process them to improve their quality and eliminate any noise that may impact the accuracy of counting. Techniques such as contrast enhancement, noise reduction, and image normalization are used to achieve this. The next stage involves segmenting the image to separate the blood cells from the background by thresholding. The segmentation process entails separating the objects of interest from the background. This process can be done using various methods such as thresholding, region growing, or watershed segmentation. In this project, we are using color thresholding method. After implementation, the evaluation metrics will come into the playground. The algorithm's performance will be measured using standard evaluation metrics such as manual checking. These metrics will help us determine the accuracy of the algorithm in detecting and classifying different types of blood cells.

This is the basic approach behind this project. Now, let's go into details of the coding implementation for this project. The code begins by importing the necessary libraries as OpenCV for image processing, NumPy for numerical operations, and math for mathematical functions. Then, the image is read that we want to process. The image file should be in the same directory as the Python script. The copy of the original image is created to later draw contours around the cells. The image is then converted from the RGB color space to the HSV color space. This allows us to easily segment the image based on color. HSV is an abbreviation for Hue, Saturation, and Value. It is a color representation format that is commonly used in digital image processing and computer vision. In HSV format, the hue is represented by the color of the image, the saturation by the purity of the color, and the value by the brightness of the color. The HSV color model is frequently used in image processing tasks including color segmentation, detection, and thresholding. The HSV color model, as opposed to the RGB color model, isolates color information from brightness information, making it easier to edit certain characteristics of an image's color. After converting the image into HSV format, lower and upper bounds for blue and red are created to do color thresholding. Binary thresholding and color thresholding are two types of thresholding methods. In the binary thresholding, one value from the grayscale is chosen and the thresholding is done according to that, however, color thresholding works different. In this program, two different colors are chosen as purple/blue and red to differentiate blood cells. Blue cells indicate that these cells are infected. Red cells are healthy cells. Color ranges for blue and red are defined to do color thresholding. First, lower, and upper bounds for the color are defined such as brighter blue and darker blue. Later, these values are converted to HSV format, and some extra values are added to both to define the range. Since we have the color ranges, we can do the thresholding. Color thresholding works by comparing each pixel in a picture to a certain color range. If the pixel's color falls within the defined range, it is identified as being part of the region of interest. If the pixel's color is outside the defined range, it is identified as part of the background or ignored. That’s what we do in the code. After thresholding, morphological operations are done to smooth the image, remove unnecessary noise, and increase the image quality in general. Let’s go into the details of morphological operations. Morphological opening and closing are two fundamental operations in image processing that are used to improve image quality by eliminating noise and undesirable structures while keeping the image's key characteristics. Morphological opening is a two-step process that involves image erosion and dilation. Erosion is the process of reducing the boundaries of an image, whereas dilation is the process of increasing the boundaries of an object. The opening procedure is designed to eliminate small objects and thin lines from an image that are smaller than the structural element used for erosion. Using a structural element, the opening process is achieved by first performing the erosion operation, followed by the dilation operation, to the input picture. This method is very helpful for eliminating small objects and tiny lines from images. The two steps of morphological closing involve erosion and dilation in the image. The boundaries of an object are enlarged by dilation and reduced by erosion. To close small gaps or holes within object boundaries that are smaller than the structuring element used for dilation, we use the closing operation. Applying the dilation and erosion operations on the input image using a structuring element is how the closing operation is carried out. This operation is particularly useful in closing small holes or gaps within the object boundaries. The size and shape of the features that will be removed or kept determine the structuring element that is used for opening and closing operations. The structuring element in the provided code is an ellipse of size (3,3), which is a popular option for removing small objects and closing the small gaps in the image. Overall, morphological opening and closing operations are vital tools in image processing for improving the quality of the image by removing undesired structures and noise while preserving important features. We have done morphological operations because we do not want any non-cell elements mess with our legit blood cells in the process of segmentation. After these operations, we have seen a huge increase in the accuracy of our algorithm. Now, we have one final problem that needs to be solved. Even though we have increased the accuracy of our program, there is still one big problem which is that some big cells are counted as one. Also, some cells are in the middle of mitotic division but since they are not fully divided yet, it is still counted as one. The solution is defining constraints and parameters as minimum cell size and average cell size. The idea behind this solution is that when we have overlapping cells, we make a guess about how many cells there are in this big chunk of cell. We defined minimum cell area as 200, average cell area as 650, and connected cell area as 1000. Do not forget that when the image size differs, these values also need to be changed. In the implementation of this solution, we go through every cell founded by find contours function. For each cell, we check if the area of it is bigger than the minimum cell area that we have just defined. In addition, we have defined connected cell area which indicates the maximum area to be considered as one cell. If the cell area is bigger than the connected cell area, it means that the cell is big enough that we cannot count it as one, therefore, we divide the cell area by the average cell area to find an estimation on number of cells in this big cell. This solution makes our algorithm more accurate, and it prevents the problem that very big cells are counted as one and cells in the middle of mitotic division which are not totally divided yet is counted as one.

3. Evaluation

The evaluation section inspects the efficiency and performance of the algorithm used and offers information on the outcomes obtained. Let’s investigate what we have achieved so far.

- Color Thresholding
  - Cells were divided into groups based on their color characteristics using the color thresholding technique.
  - For more accurate segmentation and better color representation, the HSV color space was chosen.
  - The algorithm successfully distinguished red and blue cells from the background by specifying the desired color ranges for these cells.
  - The thresholded binary images for red and blue cells were obtained and visualized.
- Morphological Operations
  - To the thresholded binary images, morphological opening and closing operations were applied.
  - The cells' overall shape was preserved thanks to the opening operation's removal of small noise regions.
  - Closing operations within cell boundaries successfully filled in gaps and eliminated small holes.
  - For the morphological operations, a structuring element in the form of an ellipse with a size of (3,3) was selected.
- Cell Segmentation and Counting
  - On the morphologically altered images, contour

    detection was used to pinpoint specific cells.
  - To eliminate small objects that are probably noise, the minimum cell area was established.
  - An estimate was made to determine the number of large, connected cells based on their area in relation to the average cell size.
  - For visualization and confirmation, the designated cells were drawn on the copy of the original image.
- Performance Analysis
  - Based on the precision of cell segmentation and counting, the effectiveness of the adopted methods was visually evaluated.
  - By contrasting the results of the contour-based analysis with the original image, the effectiveness of cell segmentation was visually evaluated.
  - The accuracy of cell counting was evaluated by comparing the total count with manual counting.
- Limitations and Future Improvements
  - Changes in image quality, lighting, and cell morphology may have an impact on how well the methods are working.
  - For various datasets, the color ranges and minimum cell area thresholds may need to be changed.
  - The algorithm assumes that cells are clearly separated from the background and stand out from it. The accuracy of counting could be impacted by the cells that are overlapping.
  - To increase the accuracy of cell segmentation and handle more complex scenarios, machine learning-based techniques such as deep learning models could be studied.

In summary, the program demonstrated effective cell segmentation and counting capabilities. The evaluation results indicate that the program performs well on the given dataset. However, further analysis and optimization are required to address potential limitations and improve overall performance but still the algorithm offers a strong starting point for additional study and advancement in the area of cell analysis and quantification.

4. Conclusion

In this project, we successfully created a color thresholding and morphological operations-based image processing program for cell segmentation and counting. In terms of precisely segmenting cells and estimating their numbers, the program performed well. The program does not only count the number of cells but also differentiate cells according to their colors. Blue or purple cells are considered as infected, red cells are considered as healthy.

We were able to distinguish cells from the background using the color thresholding technique. We obtained accurate cell segmentation by specifying suitable color ranges in the HSV color space. The regions matching to the cells of interest were successfully highlighted in the thresholded binary pictures that were produced.

To improve the results of the segmentation, we used morphological opening and closing processes. The gaps within the cell boundaries were filled in, noise was reduced, and cell forms were maintained. Achieving desired smoothing and refining effects required careful consideration of the structuring element used as well as the number of morphological operations iterated.

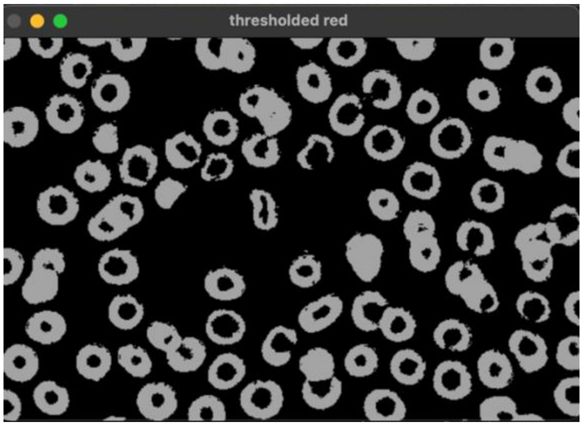
The processing of binary pictures allowed for the identification of specific cells using contour detection. In order to eliminate small items that might be noise, we created a minimum cell area threshold. By dividing the cell area by the average cell area, we calculated the number of cells inside big, connected cells. Even when there were connected or overlapping cells, this method worked well for counting and differentiating cells properly.

The evaluation of the program's performance highlighted its accuracy in cell segmentation and counting. However, it is important to consider the limitations of the program. Variations in image quality, lighting conditions, and cell morphology may impact the segmentation accuracy. Overlapping or touching cells can pose challenges to accurate cell counting. Adjustments to the color thresholding ranges and minimum cell area threshold might be required for different datasets as we have applied.

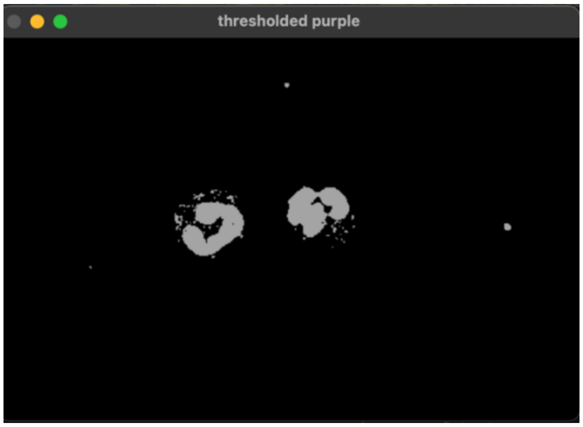
Future improvements could include incorporating noise reduction techniques, such as Gaussian blurring or median filtering, to further enhance segmentation results. Advanced techniques like deep learning-based models could be explored to handle more complex scenarios and improve accuracy.

Overall, this study offers an excellent foundation for cell analysis and quantification. Researchers in disciplines including biology, medicine, and biomedical engineering can benefit from the created image processing program. The program has the potential to assist in developing cell-based research, diagnostics, and treatments with more improvements and refinements.

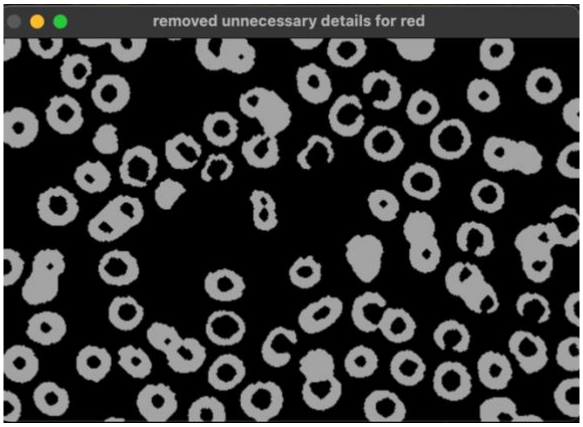
5. Images



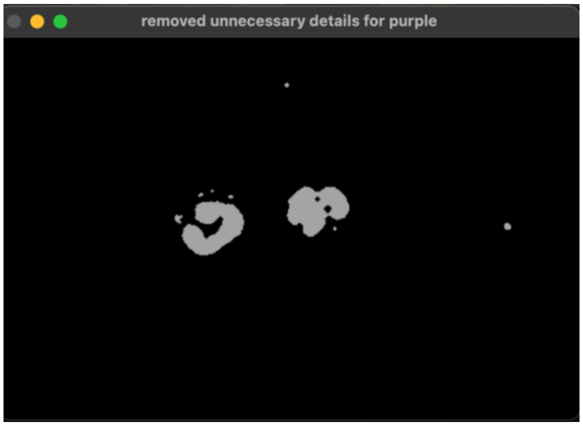
This is the output image for thresholding red cells.



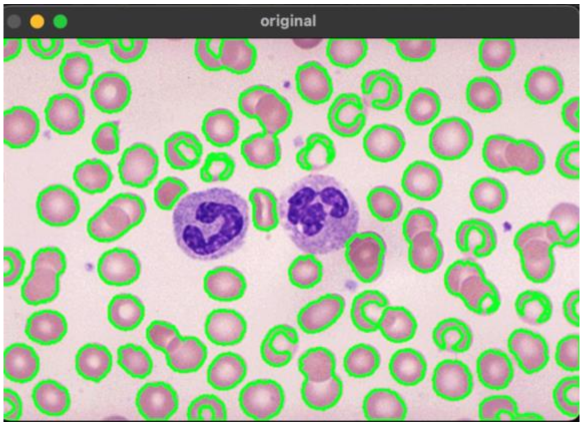
This is the output image from thresholding blue cells.



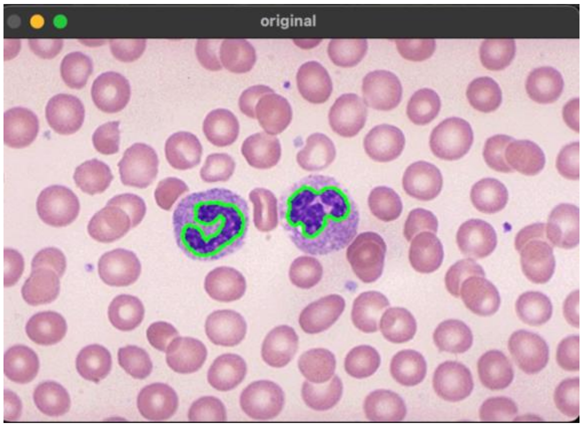
This is the output image after morphological operations on thresholded red cells.



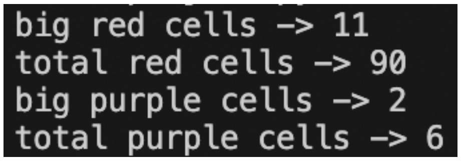
This is the output image after morphological operations on thresholded blue cells.



This is the final output image that shows identified red cells.



This is the final output image that shows identified blue cells.



```
big red cells -> 11
total red cells -> 90
big purple cells -> 2
total purple cells -> 6
```

This is the screenshot from the command line interface that shows the number of cells.