Cell Identification by Blob Detection

Mihir Gupta
Electronics & Communication Engineering (B.Tech.)
Rajasthan Technical University
Jaipur, India
mihirgupta.engg@gmail.com

***Abstract*— Medical diagnosis is expanding every day. Engineering and Medical Science are utilizing their power in a symbiotic manner, to make human life better. This paper explains and analyzes the identification of a specific microscopic cellular body using various algorithms based on Image Processing. Today, there are various instruments to identify and count microscopic cellular bodies. They make use of very thin capillaries and impedance calculation to identify and count cells. It is called Flow Cytometry. This makes them very expensive, delicate and complicated instruments. This identification can be done in another way using Blob detection, considering each cell a collection of points in same state. This state can be either color, shape, perimeter, area etc. The microscopic bodies may be Red Blood Cells (RBC), White Blood Cells (WBC), Platelets, Tumor Cells etc. The Blob detection works exactly like human eye, which differentiates on the basis of various parameters like color, shape, size, distance etc.**

Keywords— Blob, OpenCV, ROI, RBC, WBC, HSV, BGR, Threshold, Smoothing.

I. INTRODUCTION

Image Processing is rapidly expanding everyday due to its vast real time applications. One of its applications is Blob Detection. Blobs can be supposed as binary objects i.e. points that are in same state. These objects can be filtered out using various algorithms. Looking into the biological aspect of this paper, Pathology is the study of pathogens. It includes diagnosis and identification of disease causing microscopic cellular organisms.

Cellular bodies like Red Blood Cells (RBCs) and White Blood Cells (WBCs) have their own characteristic shapes and size. They can be viewed under microscopes after staining them with proper dyes. After staining, they develop a particular color, which is unique for every cell. This property differentiates them from other cellular bodies.

Blobs can be differentiated on the basis of color, shape, area, perimeter etc. An image with various shapes and colors has to undergo various processes before actual blob detection. Consider an image consisting of various cellular bodies. Among all these cellular objects, the one that needs to be identified will have its own different color, shape and area. If these parameters are put up into the program made for detection of that blob, these bodies can be separated from others in the same image.

In the present scenario the technique used is called flow cytometry, which is based on sensor technology using light and impedances. This technology is well developed and is extensively used. The demerit of this technology is its cost. Hence, developing Cell Identification by Blob Detection can be very much advantageous. It will not only cut the expenses but also open new windows for artificial intelligence.

II. BLOB DETECTION

Blob detection can be done using various platforms like MATLAB or OpenCV. In this paper, the blob detection analysis has been performed using OpenCV and the coding is done in C++. For input of data, live streaming from camera is performed. This is done by placing the image of the specimen in front of the camera or live streaming on the specimen using microscopic camera.

After the input of the data, the image undergoes various processes and transformation to obtain the required result. There processes or transformations include flipping, reading the color space, thresholding, smoothing, filtering etc. These processes are done in order to make the blob detectable.

III. INPUT STREAMING

The input for the processing of the image can be taken using the pre-installed webcam in the laptop or through any other source such as microscopic camera mounted on a trinocular. The source basically needs live streaming of the image or the tissue section itself.

If the input method is a simple camera without microscopic capabilities, the image of the tissue section on the slide must be taken. This image of the tissue should be enlarged enough and must be of good resolution. After taking the print of this image of the tissue section, it must be held in the field of view of the camera being used. This enables the system to take live streaming of the print.

The other method is using a camera mounted on a microscope that will focus directly on the tissue section present on the slide. This special camera takes live input of the desired section of the tissue on the slide.

Either of the methods can be used for the process. In the current project, both the methods are used and illustrations are

also provided for both methods. For the former method the camera used is a pre-installed webcam of 2 Megapixel and takes 18-20 FPS videos. For the latter method, the microscope used is LaboMed CXR 3 Trinocular having digital camera, with oil immersion lens to magnify.

IV. PROCESSING & TRANSFORMATION

Blob analysis for cell detection includes various transformations and processes. After the intake of the data image as live, each frame is processed to get a continuous output. The speed of operation on each frame depends on the ability of the camera and the frames per second it takes.

i. Resizing:
In this project, after capturing a frame, it is resized according to the programmer's need. The syntax code for resizing is cvResize(src,dst,int). Within the parenthesis are variables that are replaced by 'source image', 'destination image' and 'interpolation method'. In the current project the method used is CV_INTER_AREA.

ii. Flipping:
The flipping of the frame is done in case the camera is in an odd position. The flipping can be done around x and y axis. The syntax is cvFlip(src,dst,mod), in which 'mod' is the mode of flipping i.e. around x or y axis.

iii. Change in color space:
There are various kinds of color spaces like BGR, HSV, HSL etc. In this project, HSV color space is used, hence the input of BGR is converted to HSV using the code cvCvtColor(src,dst,int). Here 'int' stands for the type of conversion made, which in this case is BGR2HSV type.

iv. Comparing and Separation:
The comparing of elements of the image (pixels) and their separation is done using cvInRange or cvInRangeS. The difference between the two is that in the former, the comparison is done of the source image with the higher and lower limits of another array; while in the latter, the comparison is done with the scalar value limits set by the programmer. In this project cvInRangeS function is used so that selection can be done in real time while running the program.

v. Smoothing:
Smoothing is necessary to reduce the noise in the image and also to remove unwanted selected elements. There are various methods for smoothing like Gaussian, Median etc. the syntax used is cvSmooth(scr,dst,int,para). Here 'int' is for the method and 'para' is for the value or the level of smoothing. The function used in this project is Median smoothing.

vi. Rendering:
After removal of noise and selection of the pixels according to set parameters, the rendering of the blob is done using the function cvRenderBlobs. The Region of Interest (ROI) of the image is set to the bounding box of the blob.

V. OUTPUT

The output of all these processes will be a real time functioning 'cell identifier'. As there are various kinds of microscopic cellular bodies in a tissue section to be analyzed, the separation and identification of the desired Cellular body is done using the described processes altogether. The parameters of the desired cellular body like HSV reading, area etc are fed either by track bars while running the program or directly into the program before the codes are compiled.

The output display windows will show the following: live streaming of the input, threshold of the currently selected frame, track bars for selecting HSV values. It is programmer's choice to make more windows for functions like smoothing, resized image etc.

The window showing the threshold of the currently running frame will show the selected cellular bodies in white and every other thing in black. Hence it is basically the identification of the desired cellular part in binary.

The window showing the live streaming of the Tissue section (Input) will have markers that will be selecting the desired cellular parts. This will be the identification of the cell in the real time fed data with real time output.

VI. ILLUSTRATIONS AND OUTPUT

Here are the final output windows displaying various contents. The selection of the windows to be displayed depends on the programmer's choice.

Experiment No. 1:

In this experiment, the device used is "LaboMed CXR 3 Trinocular" and lens used is oil immersion lens. The camera is mounted on the trinocular to get the live feed of the tissue section. The software used to capture the image is "Motic Images 2000". This software is used as a driver for the camera mounted and its function is to take live input (**Fig i**).

The image of Red Blood Cells (RBC) is used for input. This image (Fig i.) also contains two White Blood Cells (WBC) in the center. The cells are stained using dyes. In the figure the small circles are RBCs that are stained and the larger ones are WBCs.

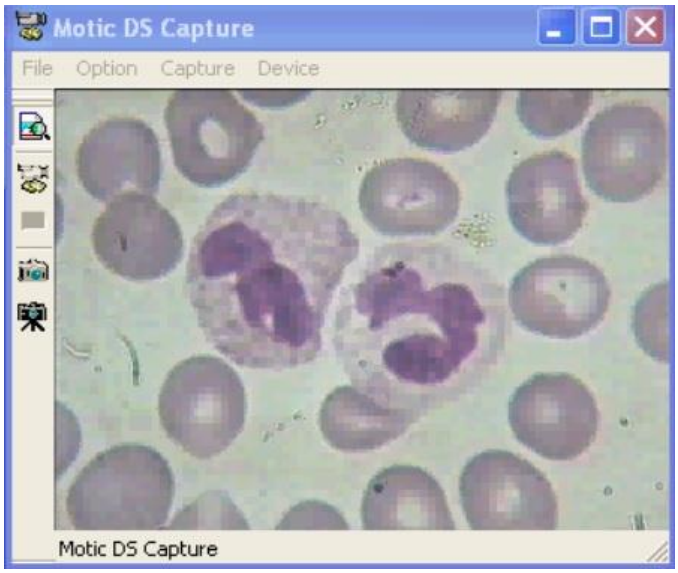

Figure i

The RBCs are small and bi-concave bodies, which have inside curved surface at top and bottom. The WBCs are larger ones with colored matter in them called nucleus.

The windows in Fig ii, iii, iv, and vi are generated using coding done in OpenCV by programmer's choice.

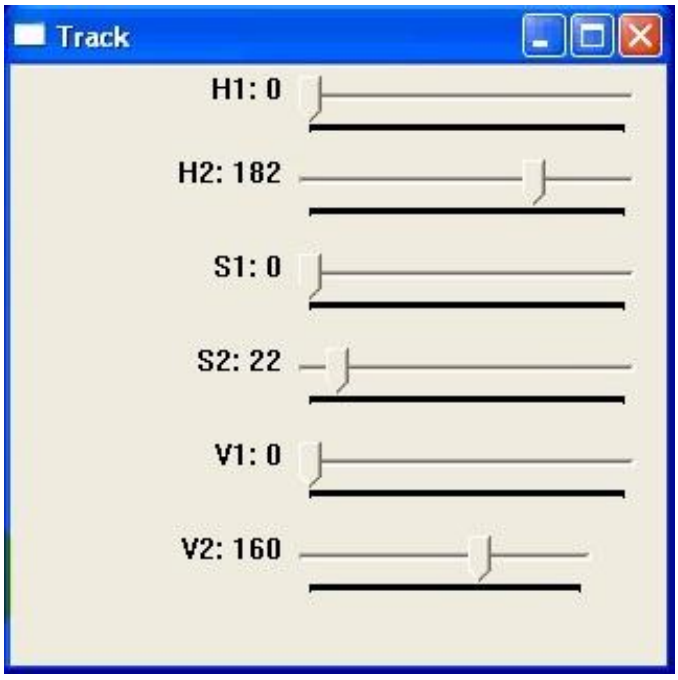

Figure ii

The next figure (**Fig ii.**) illustrates use of track bars to set Hue, Saturation and Value (HSV) parameters even while the live streaming is going on. The '1' and '2' subscripts used for HSV in track bars are for the lower and the higher limit of each. By varying them we can change the parameters for the specific cell body we want to detect.

Figure iii

The next window displays the threshold image of the input image for the selected blobs. The threshold (**Fig iii.**) displays the image in binary form i.e. white and black for 1 and 0. The blobs are displayed in white.

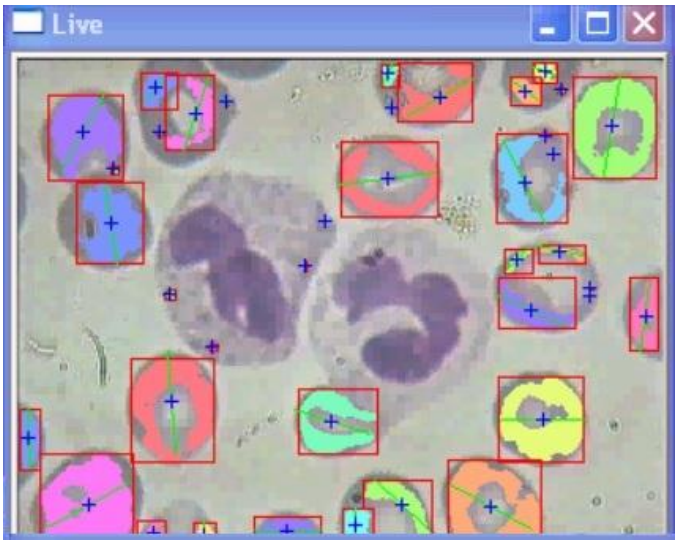

Figure iv

The final window displays the live streaming of the input along with the blobs selected. This helps in identification of the cell whose parameters are fed into the program. In this case (**Fig iv.**) the RBCs are detected and are marked and labeled. The two WBCs present in the tissue section remain undetected as their parameters do not match with those of RBC. This is the basis of cell detection using Image Processing.

Though the detection is not perfect as a single RBC is detected multiple times, the algorithm can be improvised and made to

work better according to desired environment and lighting conditions.

Experiment No. 2:

In the second experiment, the input is taken using a simple laptop camera (2MP, 18-20 fps). For input of data, an image of tissue section is used (**Fig. v**). In this case too, there are 2 WBCs surrounded by several RBCs. The image is held manually in front of the camera and the input is taken.

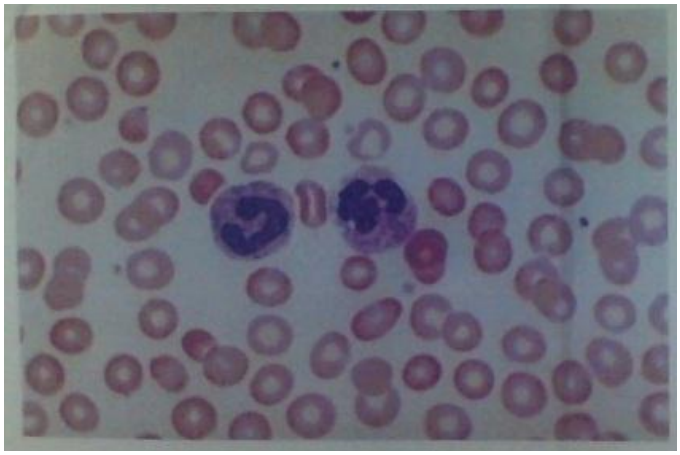

Figure v

The output in this experiment is illustrated in (**Fig. vi**), in which the RBCs are detected leaving behind WBCs undetected. Exactly like previous case, due to imperfect algorithm, all RBCs are not detected properly.

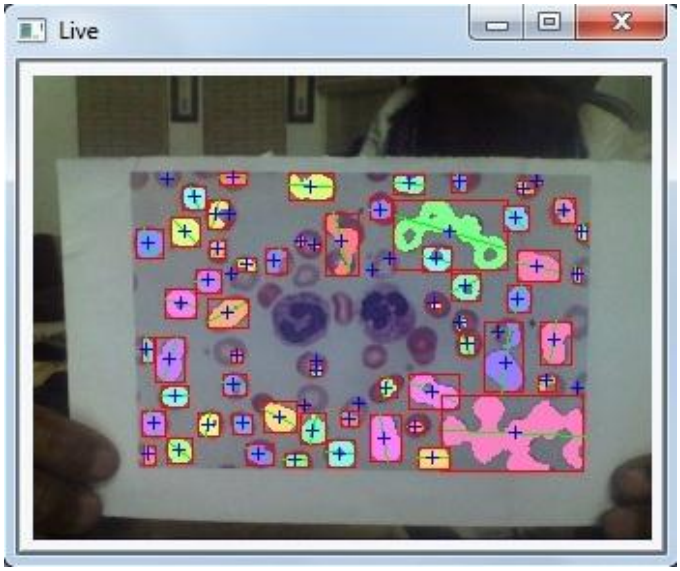

Figure vi

There are a few RBC left undetected; the reason being crude algorithm and lighting conditions. Blob detection highly depends on the lighting conditions.
This forms the basis of Cell Detection by Blob Detection.

CONCLUSION

Therefore, from this project even with crude algorithm, most of the RBCs were detected and the unwanted WBCs were left undetected. The project is currently being improvised for differentiating cells on the basis of other parameters to make the algorithm better. The accuracy is calculated for experiment no. 2.

Total no. of desired detectable cells	Total no. of cells actually detected as Blob	Percentage Accuracy (%)
88	66	75%

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my guide and teacher Mr. Chinni Krishna, for his support and help in the project. I would also like to thank my father, Dr M. L. Gupta (M.D. Pathology), for his guidance, support and encouragement.

REFERENCE

[1] Digital Image Processing - Gonzalez, Woods.
[2] Learning OpenCV - Gary Bradski and Adrian Kaehler.
[3] OpenCV 2 Computer Vision - Robert Laganière.