CORNEALULCER: A TOOL FOR SEGMENTATION AND MEASUREMENT OF CORNEAL ULCERS AND EPITHELIAL DEFECTS

Ying Wang
December 7, 2015

1 INTRODUCTION

This tool software aims at tracking corneal ulcers and epithelial defects and measuring the size. However, the current method is to measure the corneal ulcers by hand, which could become a time consuming process. Basically, there are two key points in the process:

1. Because of many unstable factors in the input images, such as different environment light conditions and angles when the patients take the photos, it's difficult to decide a universal threshold value for each input image. There are some situations when the photos are overexposed and other situations when the eyeball reflects strong flashlight from the camera, which make the problem complicated and unclear.

2. The measurement of the corneal ulcers' size is also a multi-factor problem. This is mainly because the shapes are usually irregular. So deciding how to measure regions of interests is another key point remained to be solved.

Based on the facts mentioned above, constructing an automatic segmentation system becomes hard to implement. So the main idea of the software is to provide the user a bunch of geometry process methods so they can adjust many parameters until they reach the optimal segmentation result.

2 SOFTWARE OVERVIEW

The main interface of the tool is shown in Figure 1, 2, 3. The tool takes an image as input data (in .jpg/.bmp/.jpeg format). Users can apply several geometric processing in 2D and get the cornea ulcer segmentation and a coarse measurement of it.

The interface consists of five parts: the top-left part is the slide bar window and it shows eight different parameters and let the user change their values and compare the results in the bottom result window part. The top-right part is the terminal window which is the system command line window. We can input our images, see the measurement result and quit the application through terminal window. The bottom-right part shows the original image with two bounding rectangles of the object among it, one is non-rotated rectangle, the other is rotated rectangle bounding box. The middle part shows the binary image we get after tons of operations. And the bottom-left part shows the biggest contour of our object.

3 IMPLEMENTATION

The implementation methods of the software include two parts: GUI and geometric processing algorithms.

3.1 GUI

For the GUI, I wanted to make it as simple as possible. So I built the GUI by the built-in user interface functions provided by OpenCV library. The GUI has a basic interface where users can adjust eight parameters to achieve the best results.

3.2 ALGORITHM

The easiest way to detect and segment an object from an image is the color based methods. Since the colors in the object and the background have a significant color difference based on the cornea ulcers images we have. So we can segment objects successfully using color based methods. The algorithm for tracking the corneal ulcers and epithelial defects contains the following geometry processing steps:

1. Changing Colorspaces. There are more than 150 color-space conversion methods available in OpenCV. But I am only interested in two methods which are also the most widely used ones, BGR to Gray and BGR to HSV. In the previous exercises we did in class, we often used BGR to Gray method. However, in this specific problem we can see the object we want to track has green color in most images. Usually, one can think that BGR color space is more suitable for color based segmentation. But HSV color space is the most suitable color space for color based image segmentation. So in my application I have converted the color space of original input image from BGR to HSV image.

2. Threshold Using HSV. In HSV, it is more easier to represent a color than in BGR color-space. In my application, I want to threshold the HSV image for a range of green color. HSV color space is consists of 3 matrices, "hue", "saturation" and "value". In OpenCV, value range for "hue", "saturation" and "value" are respectively 0-179, 0-255 and 0-255. "Hue" represents the color, "Saturation" represents the amount to which that respective color is mixed with white and "value" represents the amount to which that respective color is mixed with black.

In my application, I have set up the default range of green color. However, the "hue" is unique for the specific color distribution of the object and "saturation" and "value" may be vary according to the lighting condition of the environment. We only have the knowledge of approximate range values of a certain color. So the result might be far from what we expect. So I provide the user six slide bars to adjust the range of HSV. When the result meets our expectation, we can threshold the HSV image into binary image in which the object is assigned "255", otherwise "0".

3. Morphological Operation. The software provides erosion and dilation operation on the binary images. The user can drag the slide bar to choose the times of erosion and dilation. It can improve the result in some circumstances.

4. Find Biggest Contour. Each object can be considered as a contour, and the largest object can be find out by comparing area of each object. So as a first step

we have to go through each contour for comparing its area with the previous one. OpenCV provides various functions to do that.

5. Measurement Using Bounding Boxes. After we get the contour information, we can calculate the bounding rectangle of the contour object. There are two types of bounding rectangles, straight bounding rectangle and rotated rectangle. Considering the minimum area of contour, I think the rotated rectangle can give a more accurate measurement. Finally, the width and height values are exactly the data we want to know.

4 Result

Several examples' results are shown in Figure 1, 2, 3. We can notice that the segmentation result is fairly good but the measurement is unstable in some situation. This is because the bounding box of the contour sometimes can not represent the actual size of the object. It is an approximate measurement of the actual object and there are errors between them. After applying in all the images we have, I find the software can segment most images correctly except for the following three images: Cornea_B3_Fluoro3_overexposed and CorneaC_1 and CorneaC_2. The H value of the overexposed object doesn't belong to the range of 0-180 so no matter how I change the H value, it won't extract the object out of the background. And as for the CorneaC_1 and CorneaC_2, the color of the object is similar to the skin. However the skin area is much bigger than the object. So when I extract the biggest contour among the image, I can't track the object. In future, I should add some functions such as choosing the ROI to avoid those situations.

5 Future work

Due to the limited time, there are still some ideas remained to be implemented.

The first one is to reduce the users' operation. In this application, the user are still required to drag the slide bar to get a better result. In future, we can try some methods to simplify the operation as much as possible.

The second one is to improve the accuracy of the automatic measurement of the application. Now we can only get a rough calculation of the size, which means we could only know whether the corneal ulcer becomes bigger or smaller based on the change of size, but we can hardly know the accurate length and width of them. In future, we can make some tests about the relationship between the pixel value and the actual measurement of the object.

The third one is to write a better and more user-friendly GUI. At first, I planned to write up the application using Qt and OpenCV. However, something went wrong when I linked the OpenCV library with the Qt Creator. The first version I wrote is only based on Qt and I found that Qt has advantage in its neat and elegant interface. The second version I wrote is only based on OpenCV so the GUI is too simple and sometimes it is hard to use. As a result, in future I will combine them together to make this application convenient to use.

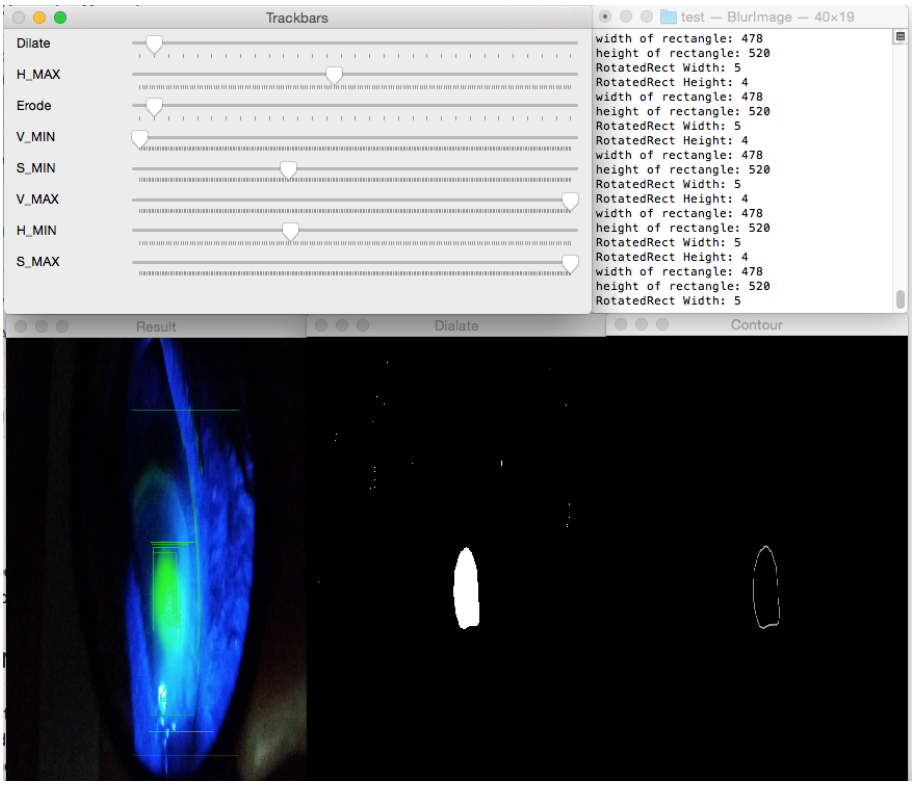

Figure 1: Cornea_B1 Result

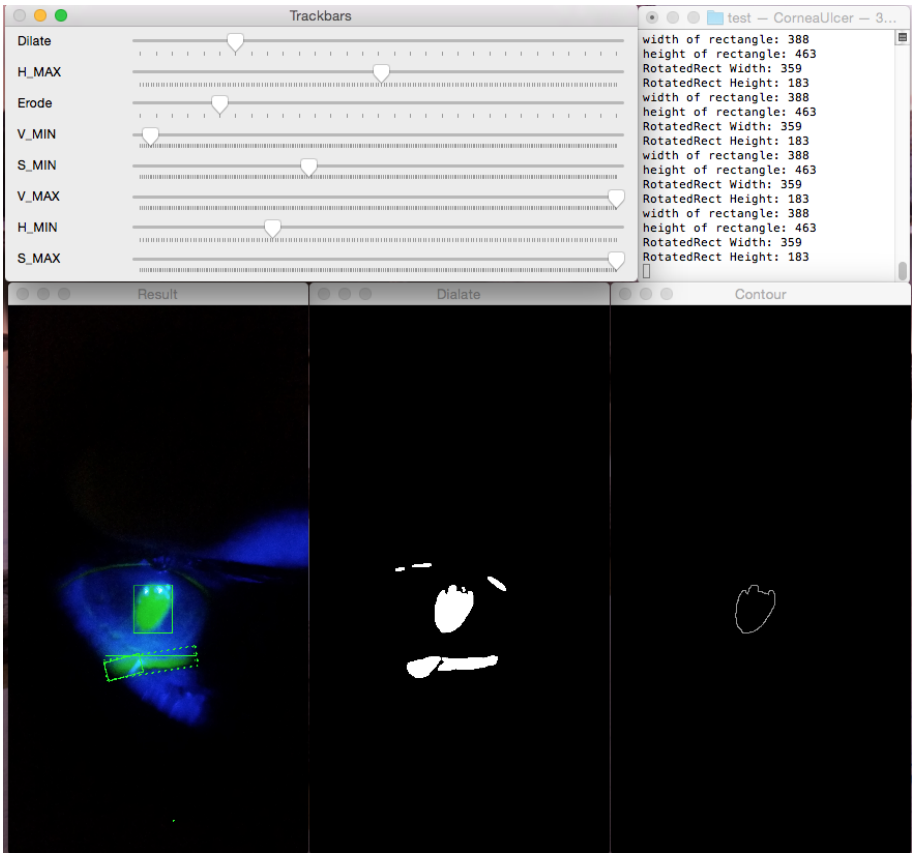

Figure 2: Cornea_B2_1_with_filter Result

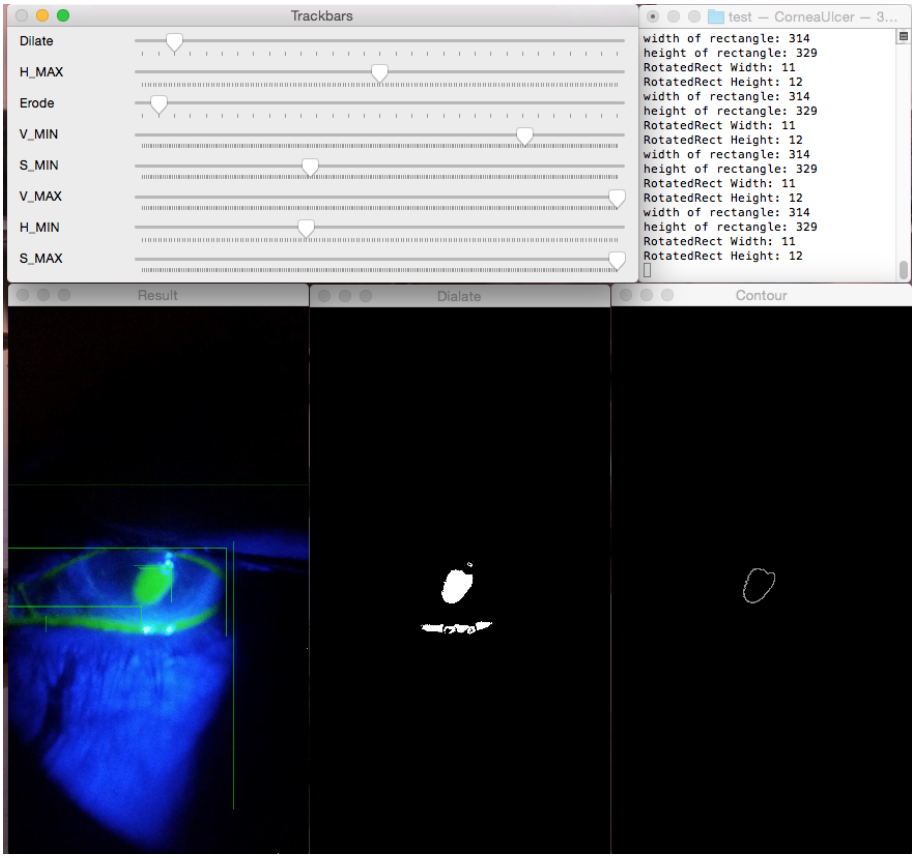

Figure 3: Cornea_B2_2_with_filer Result