# DELHI TECHNOLOGICAL UNIVERSITY

## Eye Cataract Detection using Digital Image Processing


## Digital Image Processing(EC357)

**Submitted by:-Sandeep Yadav(2K16/IT/099)**

**Satyajeet Singh(2K16/IT/103)**

# Digital Image Processing
## Project Report

**Abstract:**
This paper proposes and evaluates an algorithm to automatically detect the cataracts from color images in adult human subjects. Currently, methods available for cataract detection are based on the use of either fundus camera or Digital Single-Lens Reflex (DSLR) camera; both are very expensive. The main motive behind this work is to develop an inexpensive, robust and convenient algorithm which in conjugation with suitable devices will be able to diagnose the presence of cataract from the true color images of an eye. An algorithm is proposed for cataract screening based on texture features: uniformity, intensity and standard deviation. These features are first computed and mapped with diagnostic opinion by the eye expert to define the basic threshold of screening system and later tested on real subjects in an eye clinic. Finally, a tele-ophthamology model using our proposed system has been suggested, which confirms the telemedicine application of the proposed system.

**Robust and Efficient Automated Cataract Detection Algorithm:-**
For the automatic detection of cataract from a digital eye image, extraction of the pupil from the image is required. The pupil of an eye has the same color in all humans, only the iris color differs from person to person. The pupil is circular in shape, so we can easily detect it using common circular region detection algorithms for which Hough transform is the most common choice. It is used here to automatically detect the circular pupil, which is distinguishable. The basic method proposed in this paper for robust cataract detection algorithm can be described in three steps: preprocessing, feature extraction, and decision making. Preprocessing includes conservative smoothing followed by image denoising. The isotropic Gaussian filter is widely used as a low-pass filter for image de-noising [16]. A two dimensional (2D) Gaussian function is simply the product of two 1D Gaussian functions (one for each direction: horizontal and vertical) and is given by:

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

where ‘sigma’ is the standard deviation of the distribution and its value has been chosen as 0.5 in this work. After denoising, edge information will be lost hence it is necessary to preserve the edge. The most common and widely used approach for edge detection is canny edge detection.

The smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get the first derivative (Gx) and (Gy) . The edge gradient can be determined from these two images and the direction foreach pixel is as follows:

$$\text{Edge Gradient } (G) = \sqrt{{G_x}^2 + {G_y}^2}$$

$$\text{Angle } (\theta) = \tan^{-1}\left(\frac{G_x}{G_y}\right)$$

Gradient direction is always perpendicular to the edges. Hysteresis thresholding stage decides the real edges. After these steps, contrast enhancement is performed. These steps are followed by Histogram equalization for image enhancement. The basic idea behind this technique is mapping the intensity levels based on probability distribution of the input intensity levels.

Feature extraction is done after preprocessing to extract all the information for cataract detection and grading from the circular pupil region. The proposed detection algorithm is based on finding the accurate thresholds of texture feature parameters such as image intensity (I), uniformity (U) standard deviation (s), to distinguish between healthy and abnormal eyes. In cataract eyes, the whitish color originates from the lens region so it can be easily concluded that cataract eyes have higher intensities than normal eyes. The equation to measure mean intensity (m) is expressed as:

$$m = \sum_{i=0}^{L-1}\left(\frac{I\,(i)}{N}\right)$$

Here ‘m’ is average of intensity, ‘I’ is possible intensity, ‘N’ is number of pixel in an image, and ‘L’ is the value of possible intensity levels. Smooth and coarse textures are found in the eye. If clouding is present, a coarse texture will be observed which shows the presence of a cataract, with degree of severity depending upon the amount of clouding. Uniformity would be maximum when all gray levels have equal values. A healthy eye will show a smooth texture with higher value of uniformity in contrast to coarse texture. Uniformity can be calculated by the given equation:

$$\mathrm{U} = \sum_{i=0}^{L-1}\left(\frac{H\,(i)}{N}\right)^2$$

Here ‘U’ is measure of uniformity, ‘H’ is probability histogram of the intensity levels in an image
region, and ‘N’ is the number of pixel in an image region, Let i = 0, 1, 2 . . . , L □ 1 be the corresponding histogram, and ‘L’ is the value of possible intensity. In terms of image processing, a low value of standard deviation indicates that the pixel value tends to be very close to the average value, whereas high value of standard deviation represents that the pixel values are broadly spread out over a large range of values. The formula to calculate the standard deviation is given as:

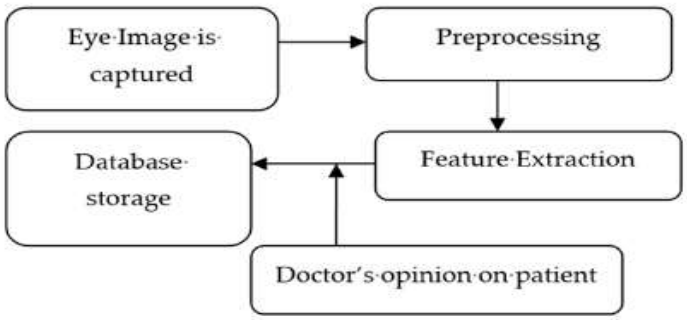
$$s = \sqrt{\frac{\sum (x_i - \overline{x})^2}{r \times c - 1}}$$

Here, ‘xi’ is the individual pixel value of the input map considered by filter, ‘x’ is the mean of the
pixel values considered by the filter, ‘r’ is the size of the filter in rows and ‘c’ is the size of the filter
in columns.
The calculation of standard deviation together with the mapping of diagnostic opinion of eye expert, adds robustness to the proposed algorithm as it helps in detecting almost exact pixel values as compared to the neighboring pixels.Decision making requires a database for comparison with the characteristic values. For the purpose of the creation of a database, diagnostic opinion of an ophthalmologist regarding the presence of cataract and its severity was recorded for 50 patients under 300 lux light intensity conditions. An image was acquired by placing the camera (Sony Cybershot, Allahabad, India) at 1entimeters distance from the eye facing forwards with camera axis almost coinciding with pupil axis. The calculation of standard deviation together with the mapping of diagnostic opinion of an eye expert adds robustness to the proposed algorithm as it helps in detecting almost exact pixel values as compared to the neighboring pixels and mapping the values with the diagnostic opinion of ophthalmologist. Comments from ophthalmologist regarding healthy, mild and severe cases were recorded. Values of texture information (mean intensity, uniformity and standard deviation) have been calculated and recorded based on an eye expert’s opinion for the classification between healthy and cataract affected eyes. Texture information and doctor’s opinion were mapped so that a threshold for detection of cataract based on texture feature (mean intensity, uniformity and standard deviation) values can be derived. Texture information

values were recorded for each case viz., healthy, mild cataract and severe cataract case as identified by a doctor. Later an individual mean value of these three cases, as per the doctor's opinion, were found using K-Means clustering technique [19]. For setting a threshold between a healthy eye and a cataract eye (both mild and severe cases), a range has been selected starting from the bottommost value recorded in a healthy eye case to the average value of K-Means cluster of two consecutive classes. The proposed system gives a decision based on this threshold value set for a normal eye and a cataract eye (both mild and severe cases together).

| Texture Information | Doctor's Opinion | | |
|---|---|---|---|
| | Healthy | Mild | Severe |
| Mean Intensity ($I$) | $12.93 < I < 125.31$ | $125.31 < I < 134$ | $I > 134$ |
| Uniformity (U) | $U > 0.17$ | $0.0927 < U < 0.11$ | $U < 0.09$ |
| Standard Deviation($s$) | $s < 1.59$ | $2.64 < s < 3.18$ | $s > 3.18$ |

Thresholding and database creation can be described by a block diagram in Figure 1.



**CODE:**

```
import os
fileName=[]
dirName=[]
for root, dirs, files in os.walk("dataset",topdown=True):
    if(len(files)!=0):
        fileName.append(files)
        #dirName.append(root.split('\\')[1])
fileName=fileName[0]
import numpy as np
import argparse
import cv2
import time

img=[]
count=0

for name in fileName:
    img.append(cv2.imread("dataset/"+name, 0));
cataract=img
```

```
    import numpy as np
    import argparse
    import cv2
    import time
N=[]

for img in normal:


        output = img.copy()
        result = img.copy()
        gray = img


        # apply GuassianBlur to reduce noise. medianBlur is also added for smoothening, reducing noise.
        gray = cv2.GaussianBlur(gray,(5,5),0);
        gray = cv2.medianBlur(gray,5)

        # Adaptive Guassian Threshold is to detect sharp edges in the Image. For more information Google it.
        gray = cv2.adaptiveThreshold(gray,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,\
                cv2.THRESH_BINARY,11,3.5)

        kernel = np.ones((2,2),np.uint8)
        gray = cv2.erode(gray,kernel,iterations = 1)
        # gray = erosion

        gray = cv2.dilate(gray,kernel,iterations = 1)
        # gray = dilation

        # get the size of the final image
        # img_size = gray.shape
        # print img_size

        # detect circles in the image
        circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, 200, param1=30, param2=45, minRadius=0, maxRadius=0)
        # print circles

        # ensure at least some circles were found
        if circles is not None:
            # convert the (x, y) coordinates and radius of the circles to integers
            circles = np.round(circles[0, :]).astype("int")

            # loop over the (x, y) coordinates and radius of the circles
            for (x, y, r) in circles:
```

```
            # loop over the (x, y) coordinates and radius of the circles
            for (x, y, r) in circles:
                # draw the circle in the output image, then draw a rectangle in the image
                # corresponding to the center of the circle
                cv2.circle(output, (x, y), r, (255, 255,255), 2)
                #cv2.rectangle(output, (x - 5, y - 5), (x + 5, y + 5), (0, 128, 255), -1)
                #time.sleep(0.5)


        # Display the resulting frame
         #   cv2.imshow('gray',gray)
         #   cv2.imshow('output',output)


        image=result[y-r:y+r, x-r:x+r]
        #cv2.imshow('Cropped Image', image)

        resized=cv2.resize(image, (256,256))

#Equalizition
        img2=resized
        equ = cv2.equalizeHist(img2)
        res = np.hstack((img2,equ))
        img1=img2
        img2=equ
        hist,bins = np.histogram(img2.flatten(),256,[0,256])
        cdf = hist.cumsum()
        cdf_normalized = cdf * hist.max()/ cdf.max()

        resultant_image=equ
        intensity=resultant_image.flatten().sum()/(256*256)
        uniformity=((hist/(256*256))**2).sum()
        N.append((intensity,uniformity))
```

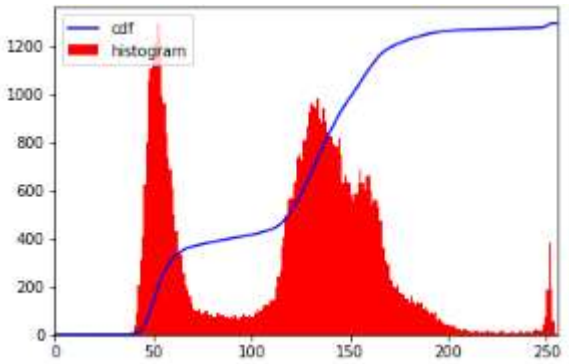
## Output:-

```
X=[]
Y=[]
for i in C:
    X.append(i)
    Y.append(1)
for i in N:
    X.append(i)
    Y.append(0)
```

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(X, Y)

print(neigh.predict([T[1]]))
```

```
[0]
```



## Process Flowchart:-

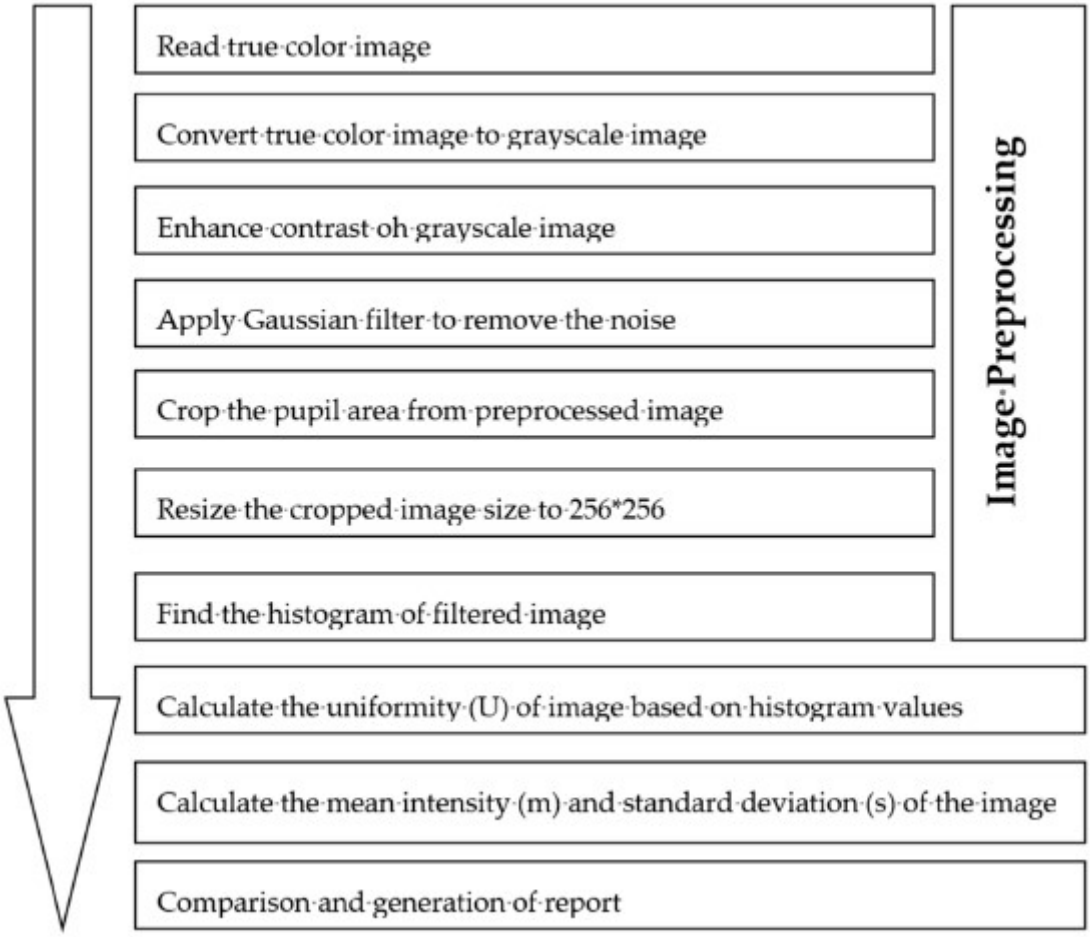



**Figure 2.** Flowchart of robust cataract detection algorithm.

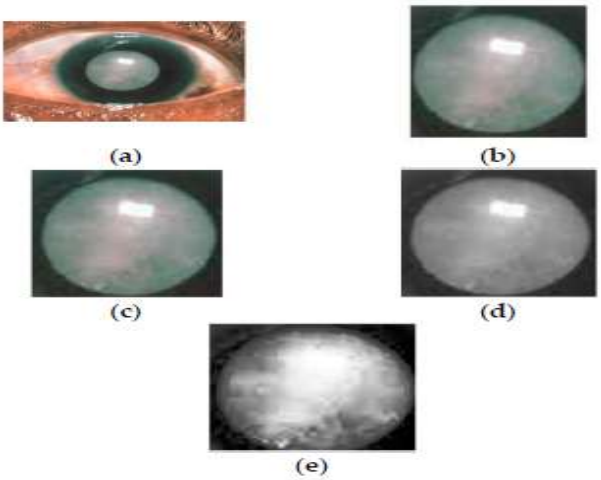



**Figure 3.** (a) Subject eye image; (b) Cropped pupil region; (c) Resized image (256 × 256); (d) Gray-scale image; (e) Contrast enhanced image.