Analysis and Processing of Biometric Images

Laboratory

Report # 2

Liu Jianyu

1. Introduction

In order to identify the iris in the picture, first, we need to use boundaries to locate eyelids, then eliminate those eyelids by thresholding, after that use Hough Transform to locate the center point of iris, detect edge of the iris, and finally draw the circle of iris.

2.

Trough thresholding, turn the image into only white and black pixels, then it's possible to pick up iris from what's left.
After upper and lower eyelids were found, we can eliminate the eyelids by thresholding then mark with NaN.

```
private void blur(){
        float ninth = 1.0f / 9.0f;
        float[] blurKernel = {
              ninth, ninth, ninth,
              ninth, 1.0f/5.0f, ninth,
              ninth, ninth, ninth,
        };
        ConvolveOp cop = new ConvolveOp(new Kernel(3, 3, blurKernel));
        BufferedImage img = new BufferedImage(bimg.getWidth(), bimg.getHeight(), BufferedImage.TYPE_INT_RGB);
        cop.filter(bimg, img);
        bimg = img;
      }

private void irisedge() {
        float[] edgeKernel = {
              0.0f, -1.0f, 0.0f,
              -1.0f, 4.0f, -1.0f,
              0.0f, -1.0f, 0.0f
        };
        ConvolveOp co = new ConvolveOp(new Kernel(3, 3, edgeKernel));
        BufferedImage img = new BufferedImage(bimg.getWidth(), bimg.getHeight(), BufferedImage.TYPE_INT_RGB);
        co.filter(bimg, img);
        bimg = img;
      }
```

3.

During Hough Transform, two circles may be found. From the original, it is obvious to say that the smaller circle would be pupil while the other is iris.

We need to find the center point of the iris and then detect the radius.

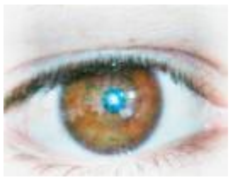
In my case, I'm not sure which method should be used to find the coordinates of center point, so I left some blanks in method *getCenterPoint(int cp)*.

```
public Point getCenterPoint(int cp) {
        ///
        /// ???
        ///
        return centerPoint[cp];
    }

    private synchronized void getImageValues() {

        int pixels[] = new int[width * height];
        PixelGrabber pixGrabber = new PixelGrabber(bimg, 0, 0, width, height, pixels, 0, width);

        try {
            pixGrabber.grabPixels();
        } catch(InterruptedException e) {
            System.out.println("Interrupted Exception: grabPixels()");
        }

        imageValues = new int[width][height];

        for(int y = 0; y < height; y++) {
            for(int x = 0; x < width; x++)
                imageValues[x][y] = pixels[y*width + x] & 0xff;
        }
    }

    private synchronized void doHoughTransform() {

            int numCirclePoints = 0;
            int min = 1;

            int maxW = width - min;
            int maxH = height - min;

            houghValues = new double[width][height];

            int numPts = Math.round((float)8 * radius);

            int circle[][] = new int[2][numPts];
            for(int i = 0; i < numPts; i++) {
                double theta = (6.2831853071795862D*(double)i)/(double)numPts;
                int xx = (int)Math.round((double)radius*Math.cos(theta));
                int yy = (int)Math.round((double)radius*Math.sin(theta));

                if((numCirclePoints == 0) | (xx != circle[0][numCirclePoints]) & (yy != circle[1][numCirclePoints])) {
                    circle[0][numCirclePoints] = xx;
                    circle[1][numCirclePoints] = yy;
                    numCirclePoints++;
                }
            }

            for(int y = min; y < maxH; y++) {
                for(int x = min; x < maxW; x++) {
                    double tempValue = sobel(x, y);

                    if(tempValue == (double)0){
                        continue;
                    }
                    for(int i = 0; i < numCirclePoints; i++) {
                        int xCoord = x + circle[0][i];
                        int yCoord = y + circle[1][i];

                        if((xCoord >= 0) & (xCoord < width) & (yCoord >= 0) & (yCoord < height)){
                            houghValues[xCoord][yCoord] += tempValue;
                        }
                    }
                }
            }
            houghPixels();
        }

        private double sobel(int xPos, int yPos) {
             double sobelSum = 0.0D;
             int d[][] = new int[3][3];

             for (int y = 0; y < 3; y++) {
                  for (int x = 0; x < 3; x++){
                    d[x][y] = imageValues[(xPos + x) - 1][(yPos + y) - 1];
                }
             }
             double sobelH = ((double)d[0][2] + (double)(2*d[1][2]) + (double)d[2][2]) - (double)d[0][0] -
(double)(2*d[1][0]) - (double)d[2][0];
```

```
			double sobelV = ((double)d[2][0] + (double)(2 * d[2][1]) + (double)d[2][2]) - (double)d[0][0] - (double)(2 * d[0][1]) - (double)d[0][2];
			sobelSum = (Math.abs(sobelH) + Math.abs(sobelV)) / (double)2;
			return sobelSum;
		}
```

4. Result

Original image – eye.jpg



Result – eyeeye.jpg

(Since I failed to get the coordinates of center point, so this is the result I assume should look like, according to the rest part of the code.)