Iris Segmentation using Geodesic Active Contours and GrabCut

Sandipan Banerjee[1] and Domingo Mery[2]

[1] Dept. of Computer Science, University of Notre Dame
[2] Dept. of Computer Science, Pontifica Universidad Catolica de Chile

Abstract. Iris segmentation is an important step in iris recognition as inaccurate segmentation often leads to faulty recognition. We propose an unsupervised, intensity based iris segmentation algorithm in this paper. The algorithm is fully automatic and can work for varied levels of occlusion, illumination and different shapes of the iris. A near central point inside the pupil is first detected using intensity based profiling of the eye image. Using that point as the center, we estimate the outer contour of the iris and the contour of the pupil using geodesic active contours, an iterative energy minimization algorithm based on the gradient of intensities. The iris region is then segmented out using both these estimations by applying an automatic version of GrabCut, an energy minimization algorithm from the graph cut family, representing the image as a Markov random field. The final result is refined using an ellipse-fitting algorithm based on the geometry of the GrabCut segmentation. To test our method, experiments were performed on 600 near infra-red eye images from the GFI database. The following features of the iris image are estimated: center and radius of the pupil and the iris. In order to evaluate the performance, we compare the features obtained by our method and the segmentation modules of three popular iris recognition systems with manual segmentation (ground truth). The results show that the proposed method performs as good as, in many cases better, when compared with these systems.

1 Introduction

Biometric features have become very popular in person identification and are being frequently used at present in defense, private organizations and government bodies. While fingerprints are still the most prevalent biometric feature, iris texture has come forth as a major player as well since the last decade or so. Its rising popularity can be gauged by the fact that iris texture information is being stored along with fingerprint of over 1 billion Indian citizens for the UIDAI project [1]. Many commercial iris recognition software has become available in the market as well. For any iris recognition system to work however, it is essential that the iris is first segmented out correctly from the eye images. But the quality of the segmentation can be drastically affected by heavy occlusion, casting of shadows, non-uniform illumination and varying shape of the iris. Examples of near infra-red (IR) eye images with varying illumination, occlusion and iris size can be seen in Figure 1. Therefore, having an efficient algorithm which can segment the iris in a near-perfect manner in all such cases is crucial.

Recognition based on iris texture came under the spotlight through John Daugman's research, where he used geometric features to segment out the iris by fitting a circle

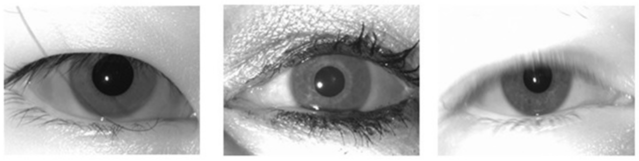

Fig. 1: Some near infra-red eye images with varying levels of illumination, occlusion and iris shape.

around it [2,3]. The relatively older iris segmentation algorithms, [4,5] being two of the popular ones, are also based on the eye geometry and work well only with images where a circular fit is possible. Consequently, noisy results are obtained from images consisting of high level of occlusion, shadows and non-circular iris shape. Some of the recent papers on the topic [7,8,9,10], which investigate iris segmentation from degraded images in visible wavelength and from distantly acquired face images, all use modern image segmentation techniques to eliminate such cases. Geodesic active contours (GAC) [6] is one such segmentation algorithm used for iris segmentation [11]. It uses intensity gradient in images to iteratively evolve an initial curve and fit it around the contour of an object via energy minimization.

The application of the concept of graph cuts in the area of image segmentation was first proposed in [13]. The image is modeled as a Markov random field and an objective function is minimized to segment the object. Histogram of the intensity of few hard-labeled foreground and background pixels marked by the user is used to categorize unlabeled pixels and minimize the objective function in a one-shot manner. Graph cuts have gained much popularity in image segmentation and have been used extensively since then [14,15]. The GrabCut algorithm was proposed to further fine-tune this technique by reducing user interaction [12]. The user draws a rectangular box around the object to specify pixels as sure background (pixels outside the box) and unlabeled pixels (inside the box). A Gaussian mixture model (GMM) is used, instead of a histogram, to categorize pixels in the box by learning parameters based on their intensity in an iterative fashion to get the segmentation. The user can additionally mark pixels inside the box as sure foreground or background to refine the segmentation (like the user marking process in graph cut). Pundlik *et al.* have discussed the use of graph cuts for iris segmentation in [16]. First, the eye image is labeled into eyelash and non-eyelash pixels using texture information with graph cuts, then they segment out the iris region from the non-eyelash pixels.

Our work uses both the GAC model and GrabCut for iris segmentation. A near central point inside the pupil is located by profiling the intensity of the near IR eye image. An initial estimation of the outer contour of the iris (the limbic boundary) and the contour of the pupil is localized by running GAC twice from the point found inside the pupil. The coordinates of the rectangular box (around the iris) and user marks (hard-labels) for sure foreground (iris) and background (pupil) pixels inside it, are computed fully automatically from these estimates without any user effort and are drawn in the image. The iris region is then segmented out from this image by running GrabCut. The segmentation is further refined by fitting an ellipse around the iris and the pupil. The contribution of our approach is twofold. First, we have developed a new robust and automatic algorithm based on GAC and GrabCut which gives an optimal iris segmentation in an iterative manner, instead of a single-shot learning approach. The robustness of our approach is comparable to that of the segmentation modules of three popular iris recognition systems (IrisBEE, VeriEye and MIRLIN). Second, the full code of our method written in Python is available for researchers to use in their own studies[3]. We have described the different steps of the algorithm, experiments and results in more detail in the following sections.

2 Proposed Method

Our method can be broadly broken down into the following steps: a) pre-processing, b) contour estimation using GAC, c) iris segmentation using GrabCut and d) a post-processing step for refinement of the segmentation using ellipse fitting. The mask with the elliptical outlines of the iris and the pupil is placed on the original image to get the final segmentation. The different steps are schematically represented in Figure 2.

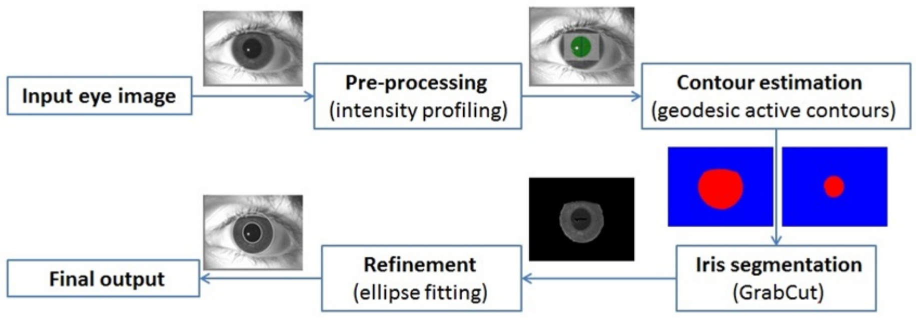

Fig. 2: Different steps of the algorithm. Intermediate images show input and output.

2.1 Pre-processing: Intensity based Profiling

The goal of the pre-processing step is to locate a central-ish point inside the pupil of the eye image. The image is first divided into three equal parts area-wise in both horizontal and vertical directions. The intersecting region at the center of the image is chosen for profiling since the pupil is generally located in the vicinity of the central region of the eye. Profiling lines are passed in this region, 5 pixels apart in both horizontal and

[3] https://github.com/sbanerj1/IrisSeg

vertical directions, capturing the intensity of each pixel they pass through. The x and y coordinates of all pixels which are within a threshold (set at 20 for our experiments) to the minimum pixel intensity, *i.e.*, the darkest pixel in the region, are stored. We get a near central point (c_x, c_y) in the pupil by computing the mean of the x and y coordinates of all such *dark* pixels (for a total of *n* pixels):

$$c_x = \frac{1}{n}\sum_{i=1}^{n} x_i, c_y = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{1}$$

Let *maxH* and *maxV* be the length (in pixels) of the longest profiling line containing *dark* pixels in horizontal and vertical directions respectively. We store the length (in pixels) of the longest line between the two, denoted as *maxL*, as shown below:

$$maxL = \begin{cases} maxH, \text{if } maxH > maxV \\ maxV, \text{otherwise} \end{cases} \tag{2}$$

So *maxL* is the length of the longest profiling line containing *dark* pixels across both directions. Figure 3 shows an example of profiling lines for a near IR eye image in our database. The coordinates of the central point (c_x, c_y) and the value of *maxL* are both used in contour estimation in the next step.

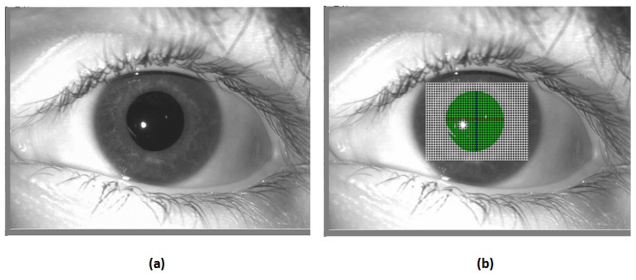

Fig. 3: (a) original eye image. (b) profiling lines are shown in white, dark pixels in green, longest lines containing dark pixels are shown in red and blue for horizontal and vertical directions respectively.

2.2 Contour Estimation: Geodesic Active Contours

Contour evolution methods are being used in computer vision research for years now, geodesic active contours (GAC) being a very prominent example. GAC tries to find a contour around the boundary of separation between two regions of the image, foreground (object) and background. This is done using the features of the content in the image (intensity, gradient). The method works by solving partial differential equations (PDE) for an embedding function which has the contour as its zero-levelset. It starts with an initial curve for detecting boundaries, denoted as γ(t), where t is the parameter for the evolution step. Let ψ be a function which captures the signed distance from the curve γ(t). $|\psi(x, y)|$, for instance, specifies the distance of the point (x, y) from γ(t).

The curve γ(t) can be termed as a level set of ψ, where a level set is a set of points which have the same value of ψ. ψ is called the embedding function of γ(t) since it implicitly embeds the evolution of the curve. So if ψ = 0, *i.e.*, the zero-th level set, then we are on the curve γ(t). This embedding function is evolved iteratively to get the contour of the object based on the intensity gradient and the morphological features of different regions in the image. But in most cases, the evolution of the curve does not stop at the boundary region because of varied intensity gradients along the edge. A geodesic term is introduced to stop curve evolution and ensure the curve is attracted towards the boundary regions in such cases, and hence the name. More details about the method can be found in [6,11].

For our work we have used the implementation[4] of morphological GAC, a variant of the traditional GAC model, which has been described in [17]. For an eye image, we run the GAC method twice with the near central point (c_x, c_y) as its center (see (1)), first to detect the outer boundary of the iris and then for the pupillary boundary. In the first case the diameter of the initial curve is set a few pixels (60 for the set of images we used) more than *maxL* (see (2)). For the pupil, the diameter is set as 0.6 $\times$ *maxL*. We set a hard limit on the number of iterations, 120 for the iris and 50 iterations for the pupil, before the evolution stops. The result of the GAC method for estimation of outer boundary of the iris and the pupil can be seen in Figure 4. The contour estimations for both the iris and the pupil are stored and used for the iris segmentation process in the next step.

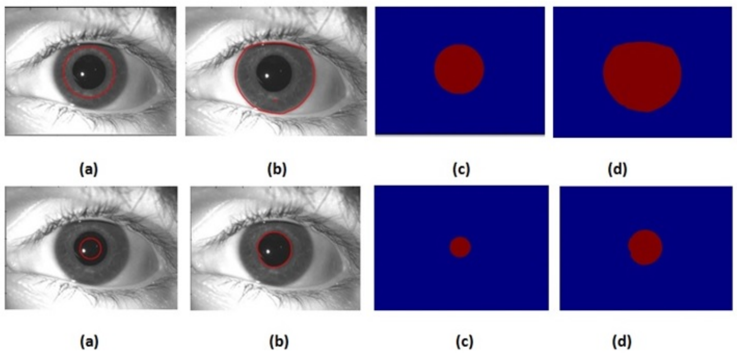

Fig. 4: Top: (a) initial iris contour (in red), (b) final contour after 120 iterations, (c) initial iris estimation, (d) final estimation. Bottom: (a) initial pupil contour (in red), (b) final contour after 50 iterations, (c) initial pupil estimation, (d) final estimation.

[4] https://github.com/pmneila/morphsnakes

2.3 Iris Segmentation: GrabCut

The segmentation model proposed in [12] is based on the same idea of energy minimization in graph cuts [13]. But instead of an one shot learning approach using greyscale intensity histograms from the hard labeled foreground and background pixels in graph cuts, Gaussian mixture models (GMM) are used in GrabCut to learn parameters for the foreground and background based on the pixel intensities in color space. The process starts with the user drawing a box around the region of interest and hard labeling some sure foreground and background pixels. The image is then modeled into a weighted graph with each pixel corresponding to a node. Two additional nodes, the source and sink terminals representing the foreground and background respectively, are also introduced in the graph. Weights are assigned to each edge formed between neighboring pixels (called *n-links*) and between a node and the source or the sink terminal (*t-links*). The graph is then disconnected into two parts using min-cut [15]. Consequently, every unlabeled pixel in the box gets categorized as either foreground or background, *i.e.*, we get a segmentation. The process continues iteratively till convergence. However, the algorithm requires some user effort due to its semi-interactive nature.

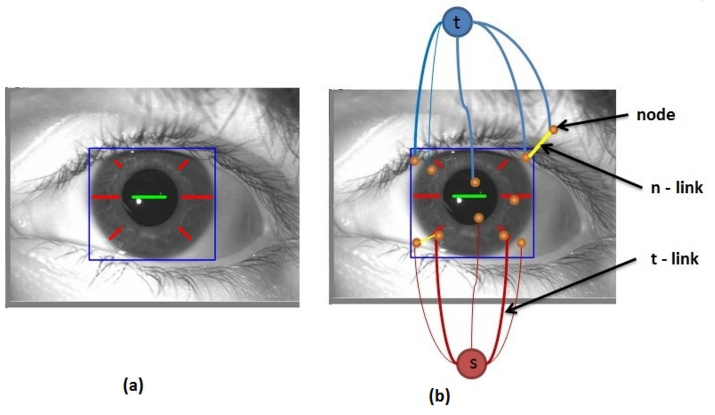

Fig. 5: (a) rectangular box containing iris, hard-labeled foreground pixels (in red), hard-labeled background pixels (in green), (b) graph formed from the image with n-links connecting nodes with neighborhood nodes and t-links connecting them with source (s) and sink (t) terminals.

We automate this box drawing and hard labeling process, using the contour estimations from GAC, so that no user effort is required. From the outer contour estimation of the iris, we locate the rightmost, leftmost, highest and lowest points on it. With the coordinates of these four points, the rectangular box around the iris is drawn on the image (after normalization, if necessary). The whole of the iris, pupil and some parts of the sclera, eyelids and eyelashes are usually inside this box. Similarly, we calculate four points in the four directions from the contour estimation of the pupillary boundary to get the location of a smaller rectangular box around the pupil. Since most pixels in the region between the two rectangular boxes belong to the iris, we label a few pixels in this region as sure-foreground, in six directions. Since the smaller box created from the pupil contour contains pixels mostly belonging to the pupil, some of them through the middle are marked as sure-background. The rectangular box and the hard label swipes generated from the contour estimations in Figure 4 can be seen in Figure 5(a). Again, this whole process is fully automatic and needs zero effort on the user's part.

The graph is then constructed from this image with the source and sink terminals (Figure 5(b)). The weights of the t-links and n-links are computed as well. Applying min-cut iteratively on this graph gives the optimal segmentation. For our set of eye images, 10 iterations generally produce a good segmentation. Most of the iris pixels are correctly labeled, as can be seen from Figure 6(a). The pixels belonging to the sclera, specular highlights, eyelids and eyelashes are removed (labeled as background) as the GMM parameters are learned from the hard-labeled pixels outside the box.

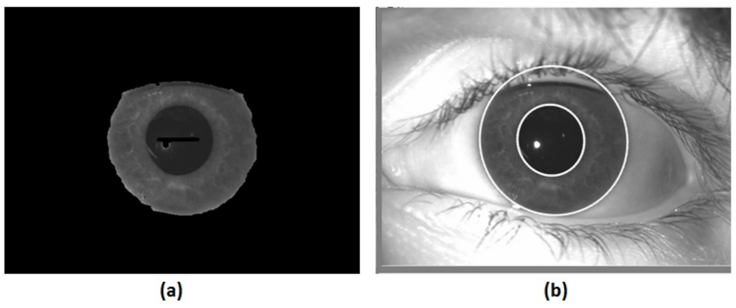

Fig. 6: (a) segmented iris after 10 iterations, (b) final segmentation after ellipse fitting.

2.4 Refinement: Ellipse Fitting

The segmented iris produced by GrabCut needs to be refined and its boundary fitted into the original eye image. To get a smooth curve from the segmentation, we use an ellipse fitting method[5] based on minimum squared error. The method tries to fit an ellipse to the set of points at the boundary region of the segmentation (the contour) which generates the minimum squared error. To do this, we first get a profile of the iris (and pupil) segmentation. This gives a closed set of points on the outer boundary of the iris (and pupil) where the ellipse is to be fit. Fitting two ellipses on to these boundary points should produce the refined iris and pupil mask.

However, occlusion due to eyelids or eyelashes can be detrimental to this ellipse fitting process. GrabCut only segments out the region of the iris which is visible, *i.e.*,

[5] http://opencv.sourcearchive.com/documentation/1.0.0-6/fitellipse_8py-source.html

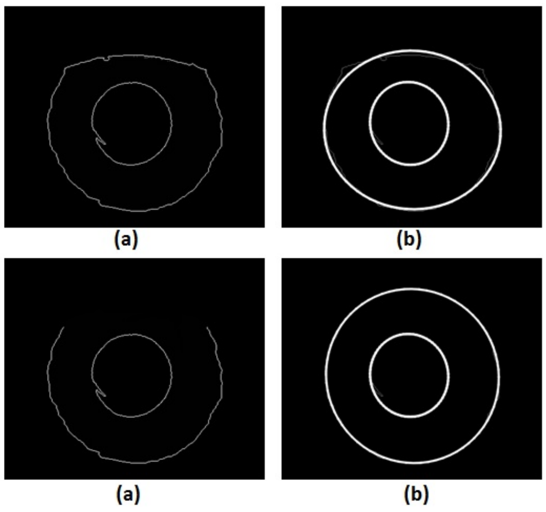

Fig. 7: Top: (a) original (closed) profile, (b) asymmetrical iris mask. Bottom: (a) open profile after removal of points in occluded region, (b) correct symmetrical iris mask.

has intensities similar to other iris pixels. So in case of occlusion, the profiling of the iris boundary segmented by GrabCut gives us a set of points which fit to the wrong ellipse. The symmetrical shape of the iris is lost in this case, as can be seen in Figure 7 (top row). To fix this problem, we check the length (in pixels) of the upper and lower tip of the segmented iris from the pupil. If one is much smaller compared to the other, below 0.75 $\times$ the larger distance for our set of images, then we conclude that part to be occluded (as the iris shape is symmetrical). In that case we remove all points in the occluded region (the smaller end from the pupil) from the profile of the iris. This produces an open set of points, with majority of the points in the curve still intact. Fitting the ellipse on to these set of points gives a much better and symmetric iris shape, as shown in Figure 7 (bottom row). Any occlusion in the pupil boundary (quite rare) gets fixed automatically in the process. The elliptical iris and pupil outlines generated, are now applied on the original eye image. This gives us the final refined segmentation (Figure 6(b)).

3 Experimental Results

In this section, we present the segmentation results obtained using the proposed method. The method was tested on the GFI database described in [18], which consists of 3000 images with both eyes of 750 males and 750 females. The images were taken with an LG 4000 sensor using near IR illumination and have a resolution of 480 (height) $\times$ 640 (width) pixels and 8-bit/pixel. For our experiments, we selected 300 left and 300 right eye images, a total of 600 images, randomly. The effectiveness of our method is shown in Figure 8 for 25 left and 25 right eye images. It is clear, that it segments the iris correctly and is able to predict the occluded part of the iris after the ellipse fitting.

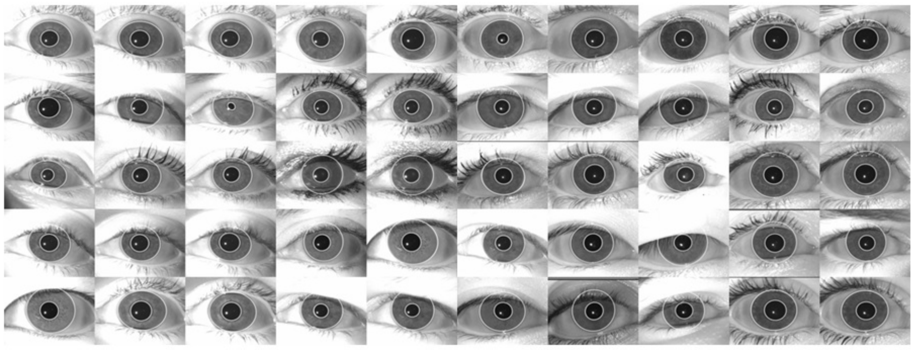

Fig. 8: Segmentation results for 25 left and 25 right eye images.

In order to evaluate the performance of our method, we tested our results against ground truth data (manual segmentation). To get a better understanding of the performance, the segmentation modules of three well known iris recognition systems were selected: IrisBEE (contour processing and Hough transform based method) [19], Neurotechnology's VeriEye SDK (active shape model based method) [20], and MIRLIN iris recognition SDK, currently owned by FotoNation [21], and their segmentation results were compared against the ground truth as well.

From all four methods (the three mentioned software and ours) and manual segmentation, we obtained two circles: one for the external boundary of the iris and the other for the pupillary boundary. From each circle, we measured the center and the radius. To get the circular radius for the iris segmented with our method, the average of the length of the two axes of the fitted ellipse was taken while the center was kept the same. We calculated the circular features for the ellipse fitted to the pupil similarly. The performance of all four methods were compared to manual segmentation by measuring the Euclidean distance between the centers (x and y coordinates) and the difference of the radius in pixels. Since the average size of the iris in these images was found to be 246×246 pixels, we normalized each difference given in pixels by dividing it by 246. The averages of all differences and distances for all methods with the ground truth was computed. The results from the comparison can be found in Table 1.

Table 1: Comparison with ground truth data

Method	pupil center	pupil radius	iris center	iris radius
IrisBee	0.55%	0.26%	1.63%	1.13%
VeriEye	0.51%	0.20%	1.52%	0.91%
MIRLIN	1.05%	0.58%	1.62%	0.69%
Proposed Method	0.49%	0.37%	1.87%	1.11%

We also extracted out the binary segmentation donut for each image for all the four methods and ground truth. We measured the precision and recall for each image by pixelwise comparison of the donuts for each method with that of the ground truth, as shown below:

$$Pr = TP/D, Re = TP/P \tag{3}$$

where *Pr* stands for precision and *Re* stands for recall. *D* and *P* are the set of positive pixels ('1') in the method segmentation and manual segmentation donuts respectively. While the intersection of the positive pixel sets of the method and manual segmentation donuts is denoted as *TP*. We further calculated the accuracy of segmentation as:

$$Acc = (TP + TN)/(P + N) \tag{4}$$

where *Acc* stands for accuracy and *N* is the set of negative pixels ('0') in the manual segmentation donut. The intersection of the negative pixel sets of the method and manual segmentation donuts is denoted as *TN*. The mean *Pr*, *Re* and *Acc* values are calculated over the 600 images for all the four methods and can be found in Table 2.

Table 2: Pixelwise comparison with ground truth data

Method	Precision	Recall	Accuracy
IrisBee	0.974	0.940	0.989
VeriEye	0.948	0.983	0.991
MIRLIN	0.948	0.975	0.990
Proposed Method	0.947	0.965	0.988

On analyzing the segmentation results of all four methods, it is found that the proposed method performs better than the other three methods in many cases, especially when there is occlusion due to eyelids or protruding eyelashes in the image. In Figure 9, the faulty segmentation results for IrisBee, MIRLIN and VeriEye can be seen on the top row and segmentation results for the corresponding images with our method on the bottom row. It can be observed that our method does a better job in these cases, especially in detecting the pupil. Furthermore, we observe that IrisBee and especially VeriEye perform consistently over the whole span of the dataset, while MIRLIN's performance varies considerably with it having the highest standard deviation for the accuracy (*Acc*) metric among all the four methods.

In this regard, it is worth mentioning that our method fails to get a perfect segmentation in some of the images, usually because the intensity distribution around the eye in those images are quite similar to that of the iris. This can be attributed to high degree of occlusion by eyelashes and heavy make-up or the images being hazy (as in the right most two images in Figure 1). As a result, optimal segmentation wasn't reached in 10 iterations of GrabCut for these images.

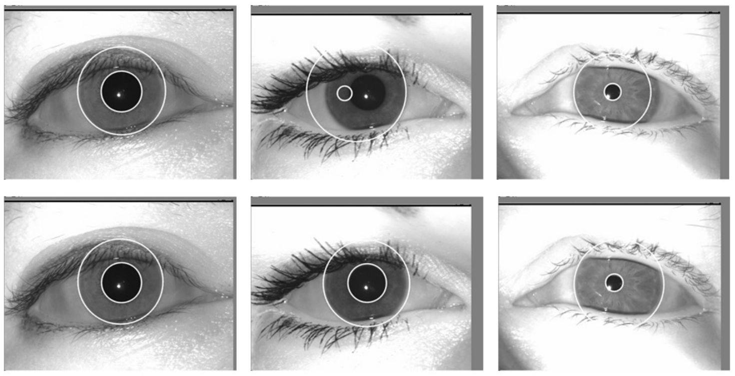
Fig. 9: Top: IrisBee, MIRLIN and VeriEye faulty segmentations respectively. Bottom: segmentation with proposed method.

4 Conclusion

In this paper, we have presented a new algorithm which performs iris segmentation automatically in cases with less constrained conditions, including variability in levels of occlusion, illumination and different locations of the iris. The main contribution of our paper is that we have developed a new robust algorithm for iris segmentation based on geodesic active contours (GAC) and GrabCut. The robustness of our algorithm is due to four reasons: *i)* a near central point inside the pupil is automatically detected using intensity based profiling of the eye image; *ii)* using that point as the center, the contours of the iris and the pupil are estimated using GAC (an iterative energy minimization algorithm based on the gradient of intensities); *iii)* the iris region is segmented using both these estimations by applying an automatic version of GrabCut (an energy minimization algorithm from the graph cut family, representing the image as a Markov random field); and *iv)* the final result is refined using an ellipse-fitting method based on the geometry of the GrabCut segmentation.

In order to validate the proposed method, we tested it on 600 near IR iris images selected randomly from the GFI dataset. To evaluate its performance, we compare the segmentation results from our method to ground truth data. The differences in locations of the centers of the pupil and iris; and the radii of the pupil and iris were less than 2%. We also obtain a 98.8% accuracy when comparing segmentation donuts pixel-by-pixel with the ground truth. To better gauge the performance, we also compare the results with that of the segmentation modules of three other popular iris recognition systems (IrisBEE, VeriEye and MIRLIN) and found them to be very identical. Since the performance of our method is found to be similar to the performance of these recognition systems, we believe that the algorithm we developed could be used by other researchers in their own studies.

Our future work involves further improving the approach in two ways. The current implementation of our algorithm in Python takes about 7 seconds for running GrabCut and 18 seconds on average for running the morphological GAC operation twice on a

Windows 7 machine with 4GB of memory. Optimizing the code will definitely speed up the process. Adding an automatic eyelash-removal step in pre-processing could give a better contour estimation in fewer iterations of GAC and GrabCut, effectively saving time. We also plan on tuning the parameters of the code and trying different combinations of them, using different ellipse fitting methods to get an idea of how they affect the performance of the system.

Acknowledgment

The authors would like to thank Juan E. Tapia for providing the segmentation results from the recognition systems, Sujoy Biswas for directing us towards GrabCut and Patrick J. Flynn for improving the paper with his suggestions.

References

1. UIDAI. https://uidai.gov.in/ 1
2. Daugman, J.: High Confidence Visual Recognition of Persons by a Test of Statistical Independence. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(11): 1148 - 1161, (1993) 2
3. Daugman, J.: Statistical Richness of Visual Phase Information: Update on Recognizing Persons by Iris Patterns. International Journal of Computer Vision, 45(1): 23 - 38, (2001) 2
4. Feng, X., Fang, C., Ding, X., Wu, Y.: Iris Localization with Dual Coarse to Fine Strategy. In: Proceedings of the IAPR International Conference on Pattern Recognition, pp. 553 - 556, (2006) 2
5. He, Z., Tan, T., Sun, Z.: Iris Localization via Pulling and Pushing. In: Proceedings of the IAPR International Conference on Pattern Recognition, pp. 366 - 369, (2006) 2
6. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic Active Contours. In: Proceedings of the International Conference on Computer Vision, pp. 694 - 699, (1995) 2, 5
7. Proenca, H.: Iris recognition: On the segmentation of degraded images acquired in the visible wavelength. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(8): 1502-1516, (2009) 2
8. Tan, T., He, Z., Sun, Z.: Efficient and robust segmentation of noisy iris images for non-cooperative iris recognition. Image and Vision Computing, 28(2): 223-230, (2010) 2
9. Tan, C. W., Kumar, A.: Unified framework for automated iris segmentation using distantly acquired face images. IEEE Transactions on Image Processing, 21(9): 4068-4079, (2012) 2
10. Tan, C.W., Kumar, A.: Towards Online Iris and Periocular Recognition under Relaxed Imaging Constraints. IEEE Transactions on Image Processing, 22(10): 3751-3765, (2013) 2
11. Shah, S., Ross, A.: Iris Segmentation using Geodesic Active Contours. IEEE Transactions on Information Forensics and Security, 4(4): 824 - 836, (2009) 2, 5
12. Rother, C., Kolmogorov, V., Blake, A.: GrabCut - Interactive Foreground Extraction using Iterated Graph Cuts. In: Proceedings of the ACM SIGGRAPH, 23(3): 309 - 314, (2004) 2, 6
13. Boykov, Y., Jolly, M.P.: Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images. In: Proceedings of the International Conference on Computer Vision, pp. 105 - 112, (2001) 2, 6
14. Boykov, Y., Veksler, O., Zabih, R.: Fast Approximate Energy Minimization via Graph Cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(11): 1222 - 1239, (2001) 2
15. Boykov, Y., Kolmogorov, V.: An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(9): 1124 - 1137, (2004) 2, 6
16. Pundlik, S., Woodard, D., Birchfield, S.: Iris Segmentation in Non-Ideal Images using Graph Cuts. Image and Vision Computing, 28(12): 1671 - 1681, (2010) 2
17. Marquez-Neila, P., Baumela, L., Alvarez, L.: A morphological approach to curvature-based evolution of curves and surfaces. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(10): 2 - 17, (2013) 5
18. Tapia, J., Perez, C., Bowyer, K. W.: Gender Classification from Iris Images using Fusion of Uniform Local Binary Patterns. In: Proceedings of the International Workshop on Soft Biometrics at ECCV, pp. 751 - 763, (2014) 8
19. Phillips, P. J., Bowyer, K. W., Flynn, P. J., Liu, X., Scruggs, W. T.: The Iris Challenge Evaluation 2005. In: Proceedings of the IEEE International Conference on Biometrics: Theory, Applications and Systems, pp. 1 - 8, (2008) 9
20. VeriEye SDK. https://www.neurotechnology.com/verieye.html 9
21. FotoNation. https://www.fotonation.com/ 9