Structured Forest Edge Detectors for Improved Eyelid and Iris Segmentation

Michael Happold

Independent Researcher
217 Hunt Rd
Fox Chapel, PA 15217, U.S.A.
mhappold@startmail.com

Abstract: Edge detection has long figured into iris segmentation algorithms, often providing a first-pass estimate of the inner and outer iris boundaries. Standard edge detectors, however, generally produce too many extraneous edges inside and outside the iris to be used for simple ellipse fitting to robustly find the iris boundaries. Solutions to this problem have been either to have additional filtering to select relevant edges or to design specialized edge detectors that highlight iris boundary edges and suppress irrelevant edge types. This description holds ever so more for eyelid boundaries, which are often very subtle. An edge detector that will trigger on an eyelid boundary will also likely trigger on almost any slight intensity gradient in an image. We seek to solve this problem by learning specialized edge detectors for each type of relevant boundary in an iris image. Using a fast Structured Random Forest approach developed for learning generalized edge detectors, we train detectors for the iris/sclera, iris/pupil, and eyelid boundaries. The results show that learned edge detectors should become part of the standard toolbox for iris segmentation and eyelid boundary detection.

1 Introduction

Edge detection seems like the obvious first step when one initially approaches the problems of iris segmentation and eyelid boundary detection. The human visual system has little trouble picking out these boundaries in all but the most degraded images. What one discovers immediately is that tweaking a threshold to find the right balance between maximizing true edges and minimizing false edges rarely produces satisfactory results and surely does not generalize to novel instances. One could search for the threshold that produces the best results over a large sample of images, but this is hardly a solution as we show in Section 6.

Better approaches than the naïve approach would be either to design specialized edge detectors for each type of boundary or to apply an independent form of filtering to the outputs of the detectors. For example, in [Hu09] the authors develop a radial-suppression edge detector that retains annular edges while suppressing radial edges by employing a non-separable wavelet transform and radial non-maxima suppression. This technique,

however, is not applicable to detecting eyelid boundaries. The Canny edge detector appears as component in numerous techniques. For example in [Ma03], the Canny edge detector is used to find prospective boundaries followed by a circular Hough transform to extract the circles best corresponding to pupil and iris boundaries. Again, this method is inapplicable to eyelid boundaries due to the assumption of a canonical shape. Pre-filtering can also be used to improve the results of edge detection, such as applying a Contourlet transform to smooth the image before employing the Canny edge detector as [ZAS12].

Daugman's integro-differential operator [Da04] has been interpreted as a circular edge detector, but it is really a circular boundary detector because the entire set of edges making up the boundary of the iris is considered when computing the gradient. Its success highlights the importance of including context in a boundary detector. Daugman extended the method to eyelid detection by changing the parameterized shapes to be second order curves with mixed results.

Similar to the iris segmentation algorithms described above, edge-detection-based techniques for eyelid boundary discovery are focused on pre- and post-filtering edge images and specialized detectors. In [Ad08], the image is filtered using anisotropic diffusion before a Canny edge detector is run, with the output then being fitted with a parabolic curve model. Another filtering step is to remove eyelashes before edge detection or parabolic curve fitting. For example, authors in [Al11] use a wavelet transform and a neural network to extract eyelashes before detecting the eyelid.

Given the stark differences between pupil, iris and eyelid boundaries, it makes sense to search for specialized detectors and to employ pre- and post-filtering on the detector outputs. However, there is a better way to design the specialized detectors than trying to make an educated guess at what is needed. Recent research on learning edge detectors can be employed to train up individualized boundary detectors that run quickly and accurately, suppressing irrelevant image gradients. They achieve this spurious edge detection by exploiting the image context of a candidate edge pixel. We make use of Structured Random Forests [Ko11] adapted to produce edge-label patches as output [DZ13] to train these specialized edge detectors. Our detectors can more accurately find the boundaries of the structures of interest than standard techniques, especially in poor quality images, such as those with significant reflections or thick eyelashes.

The rest of the paper is organized as follows. Section 2 discusses the algorithmic flow. Section 3 describes the Structured Forest approach to edge detection, starting with an overview of Random Forests and moving to the modifications needed to adapt them to structural learning. Section 4 gives an overview of the features used. Section 5 describes the training methodology. Section 6 provides results on CASIA [Ca11] and ND-IRIS-0405 [Ph06] datasets for pupil, iris, and eyelid boundary detection, comparing them to adaptive threshold Canny detection. Section 7 gives a summary, our conclusions, and suggestions for future work.

2 Algorithmic Flow

The input to our algorithm is the raw grayscale image of the eye, without cropping for the iris, taken by standard infrared IR cameras. The Structured Forest approach operates on multiple feature channels; in the original work of [DZ13], these included color channels and pixel differences. We instead apply standard edge detectors and a combination of histogram equalization and anistropic diffusion to augment the intensity image. We extract patches of these channels to include context which are then paired with corresponding label patches from hand labelings of the pupil, iris and eyelid boundaries. We then train on these pairs, producing individual classifiers for each boundary type as well as a combined classifier trained on all three boundary types. The classifiers output a label patch for a given image region representing the boundaries. Figure 1 illustrates the flow.

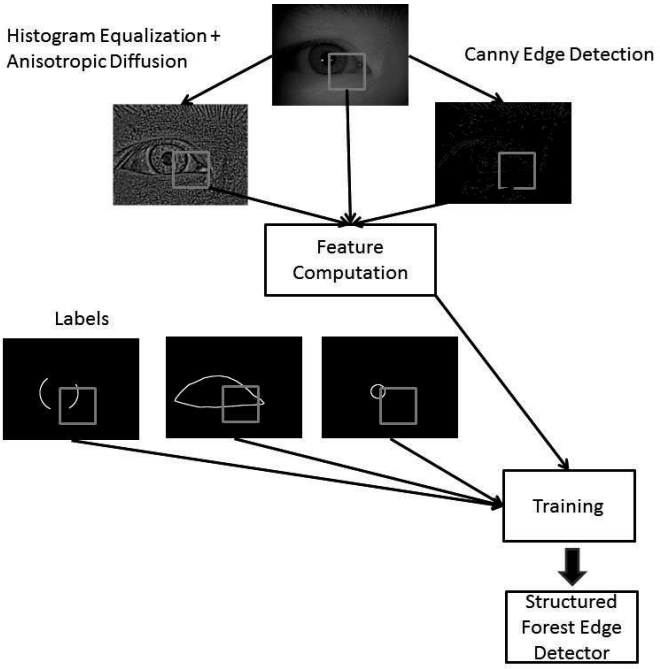

Figure 1: Algorithmic Flow

3 Structured Random Forests

The Random Forest framework forms the basis of our learning approach. A Random Forest is an ensemble of Decision Trees, each grown independently and in a manner that results in them being largely decorrelated. The output of the Forest for classification problems is a combination of the individual trees, often through simple majority voting, although other means are possible. Random Forests combine two basic ideas to increase

robustness: Bagging and random feature subset selection. Bagging [Br96], or Bootstrap aggregation, generates new training sets from a base training set by means of randomly sampling examples from the base uniformly and with replacement. By means of Bagging, each Decision Tree is exposed to a different bootstrap sample. Further robustness is introduced by randomly selecting a different subset of the feature space to grow each node of a given tree. Traditionally, the feature that maximally reduces a measure of impurity, whether Gini impurity or entropy, is chosen at each node, but randomness can be introduced at this stage as well, leading to the Exremely Random Tree. Random Forests are fast, well-suited to be adapted to structured learning tasks, and easily understood in their operations. The success of the Structured Random Forest approach of [DZ13] on the Berkeley Segmentation Dataset [Ma01] strongly suggests its applicability to iris and eyelid segmentation.

To adapt the Random Forest framework for structured learning, we need an appropriate splitting function for the tree nodes and a method for producing structured labels at the tree leaves and from the tree ensemble. Following the method of [DZ13], we treat only the output space as structured, the input space being just a bag of features. The insight of [DZ13] that the leaves of the forest‘s trees can store any type of label is key: we can simply store edge segmentation patches at the leaves that represent the edges predicted in the given image region. What this implies is that the method will only be able to predict edge masks that it has been exposed to, increasing the importance of a representative training set. This seeming limitation turns out to be less a drawback than a benefit for our domain, because the edges that we are seeking are highly constrained in their shape, relative position, and gradient orientation. For example, the boundary between the pupil and the iris is always curved in toward the pupil with a gradient that always points away from the pupil. Limiting the predicted edges to what the forest has been exposed to in the training set will work toward filtering spurious edges, particularly eyelashes and the interior structure of the iris.

Creating a splitting function for structured labels implies designing a measure of information gain, even if approximate, for a complex space. Rather than attempt to create a measure of similarity for this complex space, the Structured Random Forest approach is to map the output space Y to an intermediate space Z that is more suited to measuring the similarity between labels. The intermediate space Z should also have a straightforward mapping to the space of simple labels C, where $C = \{1,...,k\}$. Similar complex labels should at the end of the mapping have similar discrete label $c \in C$.

Our labels for edge detection in the space Y are patches of NxN binary labels, with N in this case being 16. The transformation from Y to Z presented in [DZ13] such that a Euclidean distance metric can be used in Z is to create a long binary vector of segment membership comparisons for each pair of pixels (1 if they belong to the same segment, 0 if not). There are 32640 pixel pairs for a 16x16 pixel patch, so creating the exact space Z for every member of Y would be too expensive for an edge detection algorithm. To reduce the dimensionality, the space Z is randomly sampled down to 256 dimensions, which gives an added degree of randomization, and is then further reduced to 2 dimensions via Principal Component Analysis (PCA).

The next step is to develop an information gain criterion for splitting nodes in the trees. The method of [DZ13] is used here, which uses standard information theory entropy to compute information gain. PCA discretization yields two classes, but this discretization occurs at each node during training rather than globally, and so the distributions found at each node determine different discretizations, further decorrelating the trees.

Because each leaf could have multiple label patches, we must have a method for combining them into a single prediction. Similarly, the ensemble of trees needs to combine possibly conflicting predictions from each tree. At each leaf, the medoid z_k is chosen, where z_k is defined as $\min_k = \sum_j \left(z_{kj} - \bar{z}_j\right)^2$. This same method is used to combine the outputs of the trees to produce a final label. Using the mediod is what limits the forests from generating novel results; what comes out is the already observed label that best fits the data. As we noted, this apparent limitation is in fact an important constraint in our domain.

4 Features

The work of [DZ13] focused on segmenting and detecting edges in color images; thus the features used were primarily derived from the color space, including simple pixel look-ups and color gradient magnitudes. This makes less sense in the iris segmentation domain as most iris imagery is grayscale representation only an infrared channel. There is also no reason why we shouldn't learn from low level edge information produced by standard detectors such as the Canny edge detector. Thus we replaced the color space features with those derived from the raw grayscale image, as well as from the output of a Canny edge detector and from histogram equalization followed by anisotropic diffusion. Anisotropic diffusion [PM87] applied to a histogram equalized image serves suppress noise in the image while preserving the initially faint edges, such as those found along the eyelid, that histogram equalization will strengthen. Formally, anisotropic diffusion is defined as:

$$\frac{\partial I}{\partial t} = \nabla \cdot (c(x, y, t)\nabla I),$$

where $\nabla \cdot$ is the divergence operator, $c(x, y, t)$ is the diffusion coefficient that controls its rate, and ∇I is the image gradient. In our case, the coefficient was chosen to be

$$c\left(\left\|\nabla I\right\|\right) = e^{-\left(\left\|\nabla I\right\|/K\right)^2},$$

where K is a constant that controls how sensitive the diffusion is to edges. An example of this process on a typical iris image from CASIA is found in Figure 2. A comparison with the output of Canny edge detection is shown in Figure 3, where the anistropic diffusion output is inverted, thresholded, and then small blobs are filtered out. What can be seen relative to the Canny output is suppression of the eyelashes while the eyelid boundaries in the lower left corner of the eye are picked out.

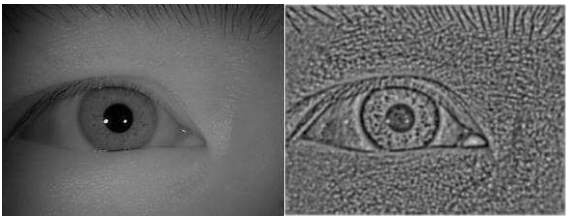

Figure 2: Histogram equalization followed by anisotropic diffusion (R) on an image (L) from the CASIA dataset

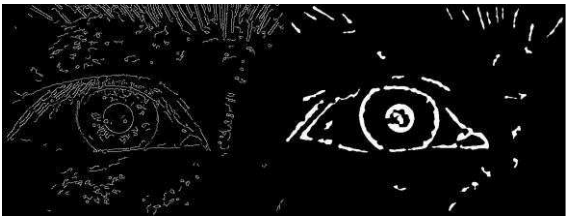

Figure 3: Comparison of Canny edge detection (L) and filtered anistropic diffusion (R). Note the suppression of eyelashes and highlighting of faint eyelid boundaries

Similar to [DZ13], we compute the gradient magnitudes at two scales and the gradient orientations, which we discretize into four bins, but we does this individually for the grayscale intensity image and the output of anistropic diffusion, producing four magnitude channels and 16 orientation features at each pixel. We add to this the raw intensity and diffused intensity values as well as the binary edge value from the Canny edge detection mask. In total, this gives us 23 features for each pixel in a patch, to which we apply the method of [DZ13] for downsampling and selecting patch features. This consists of downsampling by a factor of 2 in each patch dimension to get to 5888 features per patch. Each feature channel is blurred using a triangle blur and downsampled to 5x5, from which all pairwise differences for that channel are computed. Thus, for 23 channels, we have $23 \cdot \binom{5 \cdot 5}{2} = 6900$ additional features, for a total of 12788 for each patch.

5 Training and Testing Methodology

Ground truth edge masks for training and testing were created by hand-labeling randomly chosen subsets of the CASIA-Iris-Thousand and ND datasets with simple polygons representing the eyelid and ellipses representing the iris and pupil boundaries. The eyelid polygon is used to trim the iris and pupil ellipses when they overlap the eyelid.

We break up the labeled CASIA-Iris-Thousand dataset into a training and a testing hold-out set; we do the same for the ND dataset. We then train a classifier on the CASIA training data and test on its hold-out set, repeating the same procedure on the ND

dataset. Because cross-dataset validation is an important test of generalizability [To11], we then use the entire CASIA dataset as our training set and test on the entire ND set (and vice-versa). Our evaluation methodology is to generate Precision-Recall curves and Average Precision scores by varying the threshold at which a response from the classifier is considered an edge and checking whether there is a corresponding edge in the ground-truth edge mask. The method for determining the correspondence between a machine-generated edge and a ground-truth edge is described in detail in [Ma04]. It is formulated as a bipartite assignment problem with the distance between a machine-generated edge pixel and a ground-truth edge pixel serving as the weight between the two in the graph. Full details of this method are beyond the scope of this paper and can be found in the above reference. False positives are those machine-generated edge pixels that do not match with any ground-truth within a given distance threshold. False negatives are those ground-truth edge pixels that have no matching machine-generated edge pixels.

This training and testing pipeline is used for both a classifier trained with Canny and anisotropic diffusion features as well as one trained using only intensity as input to the algorithm. We compare both classifier types to the results from a Canny edge detector, this time varying its low threshold [Ca83] to generate Precision-Recall curves.

6 Results

On both the CASIA and ND datasets, both classifier types substantially outperform a Canny edge detector, as seen in Figure 4 and Figure 5. The Canny edge detector performance is shown in blue, our classifier using only intensity input is shown in green, and our classifier using Canny edge and anisotropic diffusion input is shown in red.

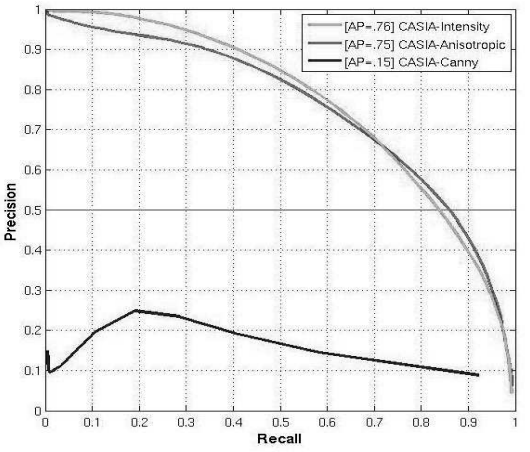

Figure 4: Precision-Recall for the CASIA dataset. Average Precisions are given in the Legend.

The classifier trained on the combination of intensity, Canny edge, and anisotropic diffusion slightly outperforms the classifier trained on intensity alone at the higher recall

levels for the CASIA set, but the difference is insignificant. An example of Canny edge detection versus the two classifier types on a CASIA image is shown in Figure 6. Note the suppression of responses to reflection edges in the learned edge detection (bottom row), and that both classifier types pick out eyelid edges in addition to finding iris and pupil boundaries.

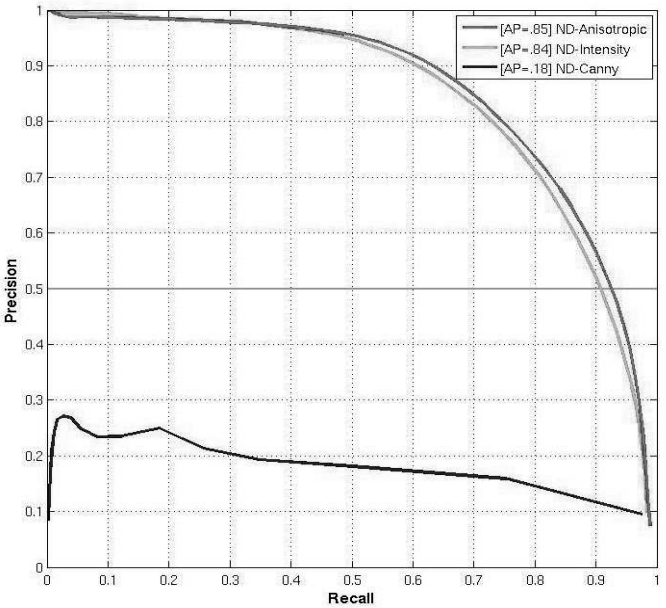

Figure 5: Precision-Recall Curve for the ND dataset

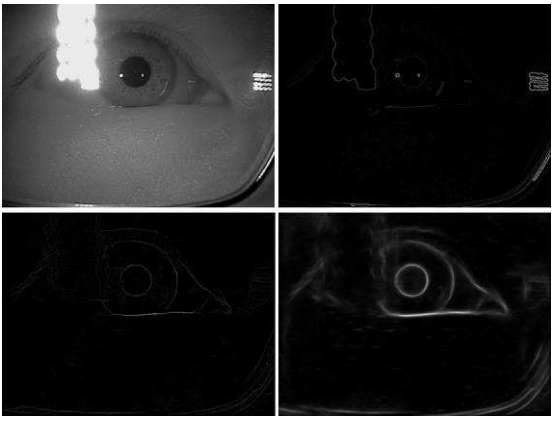

Figure 6: An example of Canny edge detection (Upper Right) and learned edge detection (Bottom Row) on the CASIA, with Intensity, Canny, and Anisotropic Diffusion inputs (Bottom Left) versus purely intensity inputs (Bottom Right).

The Precision-Recall curve for the ND dataset also shows a small gain for adding Canny edge and Anisotropic Diffusion features. An example of results on the ND set is given in Figure 7. Again, both classifiers pick out eyelid boundaries along with the pupil and iris edges while suppressing eyelash and reflection edges.

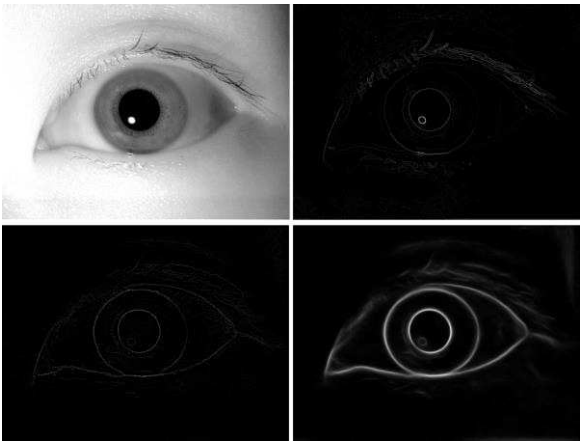

Figure 7: An example of Canny edge detection (Upper Right) and learned edge detection (Bottom Row) on the ND dataset, with Intensity, Canny, and Anisotropic Diffusion inputs (Bottom Left) versus purely intensity inputs (Bottom Right).

Our cross-dataset validation was conducted using CASIA to train and ND to test, and vice versa. There is a substantial drop-off in performance compared to the intra-dataset evaluation for both training-testing schemes, although the results still greatly exceed the accuracy of using the Canny edge detector to find these important boundaries. The benefit of using the additional anistropic diffusion and Canny inputs over purely intensity becomes evident in the cross-dataset validation.

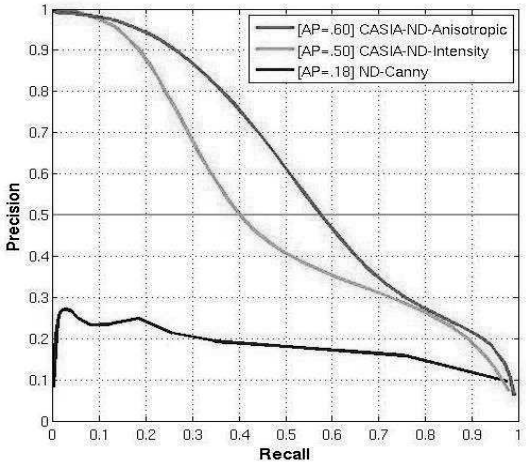

Figure 8: Precision-Recall Curve on the ND dataset for classifiers trained on the CASIA dataset

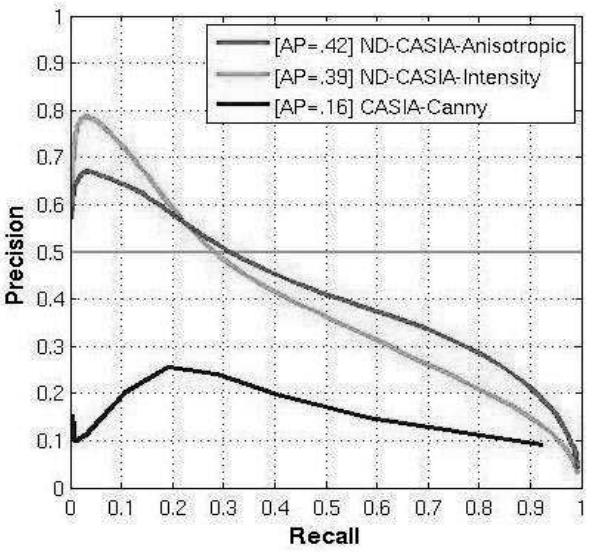

Figure 9: Precision-Recall Curve on the CASIA dataset for classifiers trained on the ND dataset

A comparison of the Average Precisions (APs) for each of the methods on the CASIA and ND datasets is given in Table 1. The two classifier types achieve APs that are 4-5 times greater than the Canny edge detector.

Table 1: Intra-dataset average precisions

Method	Training Set	Test Set (holdout)	Average Precision	Training Set	Test Set (holdout)	Average Precision
Classifier with Intensity inputs	CASIA	CASIA	0.76	ND	ND	0.3
Classifier with Canny, Anistopic, and Intensity Inputs	CASIA	CASIA	0.75	ND	ND	0.85
Canny	N/A	CASIA	0.15	N/A	ND	0.18

The AP scores for the classifiers drop significantly in the inter-dataset test, though they still outperform the Canny edge detector by 2.5 to 3.5 times, as shown in Table 2. Note that the test sets are the full labeled datasets indicated rather than just their hold-out sets used in the intra-dataset comparison, hence the slight variations in Canny AP scores between the two experimental set-ups.

Table 2: Inter-dataset average precisions

Method	Training Set	Test Set (Full)	Average Precision	Training Set	Test Set (Full)	Average Precision
Classifier with Intensity inputs	CASIA	ND	0.5	ND	CASIA	0.39
Classifier with Canny, Anistopic, and Intensity Inputs	CASIA	ND	0.6	ND	CASIA	0.42
Canny	N/A	ND	0.18	N/A	CASIA	0.16

7 Conclusion

The learned edge detectors described here significantly outperform the Canny edge detector at finding eyelid, pupil and iris boundaries while showing admirable resistance to irrelevant edges such as eyelashes. Using the boundaries generated by this methodology would remedy the weaknesses of edge-based approaches to segmenting the usable iris regions in eye imagery. There is considerable opportunity for improving even further on these results by developing additional features that generalize better across datasets.

References

[Ad08] Adam, M. et. al.: Reliable Eyelid Localization for Iris Recognition. In Proc. 10th Int. Conf. on Advanced Concepts for Intelligent Vision Problems, 2008.

[Al11] Aligholizadeh, M. et. al.: Eyelid and Eyelash Segmentation based on Wavelet Transform for Iris Recognition. In Proc. 4th Int. Congress on Image and Signal Processing. 2011: pp. 1231-1235.

[Br96] Breiman, L.: Bagging Predictors. In Machine Learning, 24(2), 1996; pp. 123-140.

[Ca83] Canny, J.: A Variational Approach to Edge Detection. In Proc. of AAAI-83, 1983; pp. 54-58.

[Ca11] CASIA Iris Image Database (ver 4.0). http://www.sinobiometric.com/casiairis.html (Retrieved 1-10-2011).

[Da04] Daugman, J.: How Iris Recognition Works. In IEEE Trans. on Circuits and Systems for Video Technology, 14(1), 2004; pp. 21-30.

[DZ13] Dollar, P.; Zitnick, C.: Structured Forests for Fast Edge Detection. In Proc. IEEE Int. Conf. on Computer Vision, Sydney, 2013; pp. 1841-1848.

[Hu09] Huang, J. et. al.: A Novel Iris Segmentation using Radial-Suppression Edge Detection. In Signal Processing, 89(12), 2009; pp. 2630-2643.

[Ko11] Kontschieder, P. et. al.: Structured Class-Labels in Random Forests for Semantic Image Labelling, In Proc. 13th Int. Conf. on Computer Vision, 2011; pp. 2190-2197.

[Ma01] Martin, D. et. al.: A Database of Human Segmented Natural Image and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In Proc. 8th Int. Conf. on Computer Vision, 2001; pp. 416-423.

[Ma03] Masek, L.: Recognition of Human Iris Patterns for Biometric Identification. M.S. Dissertation, University of Western Australia, 2003.

[Ma04] Martin, D. et. al.: Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 26, no. 5, 2004:; pp. 530-549.

[PM87] Perona, P.; Malik, J.: Scale-space and Edge Detection Using Anistropic Diffusion. Proc. IEEE Computer Society Workshop on Computer Vision, 1987; pp. 16-22.

[Ph10] Phillips, P. et. al.: FRVT 2006 and ICE 2006 Large-Scale Experimental Results. In IEEE Trans. on Pattern Analysis and Machine Intelligence, 32(5), 2010; pp. 831-846.

[To11] Torralba, A.; Efros, A.: Unbiased look at dataset bias. In Proc. 13th Int. Conf. on Computer Vision, 2011; pp. 1521-1528.

[ZAS12] Zali-Vargahan, B.; Amirani, M.; Seyedarabi, H.: Contourlet Transform for Iris Image Segmentation. In Int. Journal of Computer Applications, 60(10), 2012; pp. 41-44.