

# Выделение областей затенения радужки классификатором локальных текстурных признаков

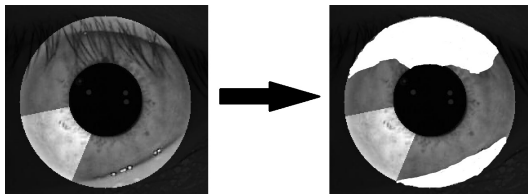
Соломатин Иван Андреевич,  
Матвеев Иван Алексеевич.

Московский физико-технический институт,  
ВЦ РАН.

23 сентября 2015 г.

## Цель:

Построить алгоритм, выделяющий область затенения радужки, использующий классификатор, основанный на смесях гауссианов, обучающийся на пикселях незатенённого сегмента радужки на том же изображении.



## Вход:

- Чёрно-белое изображение  $I$  (Матрица 1-байтовых чисел  $n \times m$ )
- $x_P, y_P, r_P$  - координаты центра и радиус окружности, аппроксимирующей границу зрачок-радужка.
- $x_I, y_I, r_I$  - координаты центра и радиус окружности, аппроксимирующей границу склера-радужка.
- Серединный угол сегмента  $S$ , не содержащего затенений.

## Выход:

Бинарная матрица  $n \times m$ , каждый пиксель которой показывает, является ли исходное изображение в соответствующем пикселе - затенением.

Более формально:

$$I \supset \Omega = \left\{ (x, y) : \begin{cases} (x - x_P)^2 + (y - y_P) \geq r_P^2 \\ (x - x_I)^2 + (y - y_I) \leq r_I^2 \end{cases} \right\} - \text{кольцевая}$$

область локализации радужки.

Необходимо классифицировать все пиксели из  $\Omega$  на два класса, т.е. построить классификатор:

$$Q(x, y) : \Omega \rightarrow \{0, 1\}. \quad Q(x, y) = \begin{cases} 1, & \text{затенение} \\ 0, & \text{радужка} \end{cases}$$

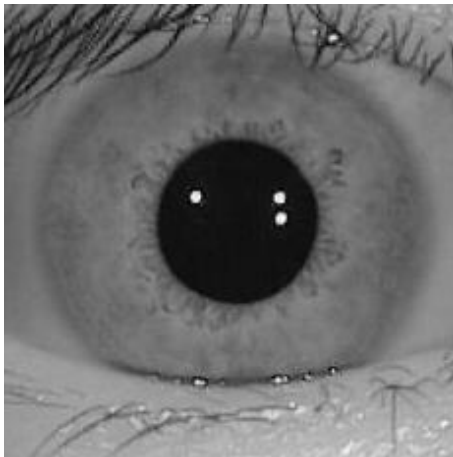
- 1 J. Daugman, *How iris recognition works*, ICIP (1). 2002. pp. 33-36.  
В данной работе граница век выделяется с помощью интегрально-дифференциальных операторов.
- 2 J. Daugman, *New methods in iris recognition*, Systems, Man, and Cybernetics, Part B, IEEE Transactions on, vol. 37, no. 5, pp. 1167–1175, Oct. 2007.  
В данной работе используется **метод активного контура**, с помощью которого определяется **граница век**.

- ③ D. Zhang, D.M. Monro, and S. Rakshit, *Eyelash removal method for human iris recognition*, Image Processing, 2006 IEEE International Conference on, pp. 285–288, Oct. 2006.  
В данной работе используется **фильтр Собеля** для **определения ресниц**. После они удаляются с помощью **медианного фильтра**.
  
- ④ Yung hui Li and Marios Savvides. *A pixel-wise, learning-based approach for occlusion estimation of iris images in polar domain*. In ICASSP, pages 1357-1360. IEEE, 2009.  
В данной работе также реализован метод поиска затенений с помощью **классификатора, основанного на смесях гауссианов**, однако, для его обучения **используется ручная разметка**.

# Метод решения

Метод состоит из следующих этапов:

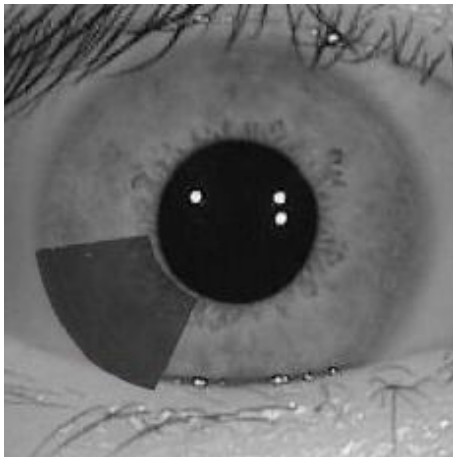
- 1 На вход подаётся изображение, кольцевая область  $\Omega$  и затенённый сегмент  $S$ .



# Метод решения

Метод состоит из следующих этапов:

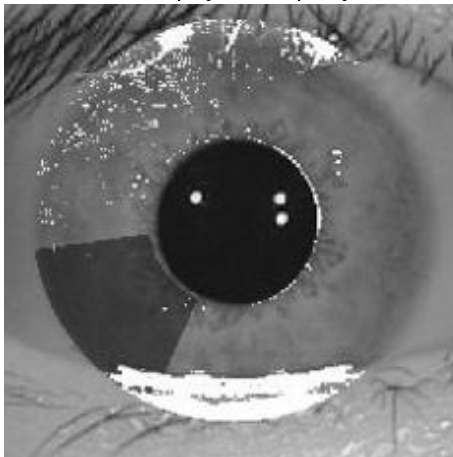
- 2 Строится классификатор, основанный на смесях гауссианов, который обучается на пикселях незатенённого сегмента.





Метод состоит из следующих этапов:

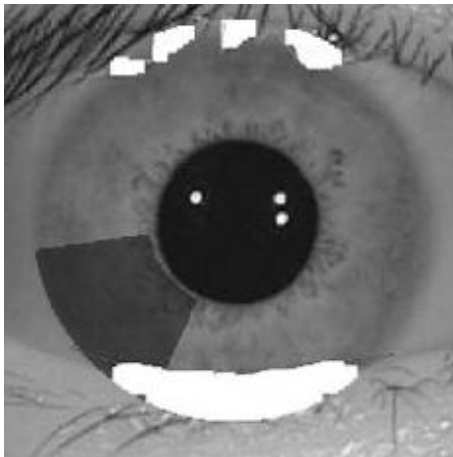
- 3 Все точки, лежащие внутри кольцевой области радужки, с помощью классификатора, относятся к тому или иному классу. Все результаты сводятся в бинарную матрицу  $J$ .



# Метод решения

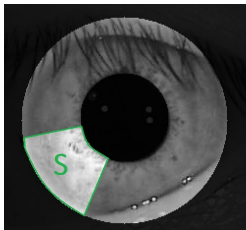
Метод состоит из следующих этапов:

- 4 Полученная бинарная матрица  $J$  подвергается постобработке с помощью операций математической морфологии.



Более подробно.

Для построения классификатора  $Q$ , рассмотрим точки, находящиеся внутри незатенённого сегмента  $S = \{\vec{x}_n\}_{n=1}^N$ .



Локальные текстурные признаки этих точек будут являться обучающей выборкой для классификатора. В качестве локальных текстурных признаков выбираем  $K = 8$  признаков.

- $B(\vec{x}_n)$  - Яркость в точке  $\vec{x}_n = (x, y)^T$
- $\bar{B}(\vec{x}_n)$  - Средняя яркость в окрестности точки  $\vec{x}_n$ .  
(Окрестность - квадрат со стороной  $a$ )
- $\sigma(\vec{x}_n)$  - Среднеквадратичное отклонение яркости в окрестности  $\vec{x}_n$ . (Окрестность - квадрат со стороной  $b$ )
- $\vec{C}(\vec{x}_n)$  - вектор из пяти компонент дискретного косинусного преобразования в точке  $\vec{x}_n$ . ДКП считается в окне  $7 \times 7$ .

Результат двумерного ДКП - матрица коэффициентов  $\hat{C}$  размерности  $7 \times 7$ . Из  $\hat{C}$  выбираем первые 5 коэффициентов за исключением  $\hat{C}_{0,0}$ :

|       |       |       |       |     |
|-------|-------|-------|-------|-----|
| (0,0) | (1,0) | (2,0) | (3,0) | ... |
| (0,1) | (1,1) | (2,1) | (3,1) | ... |
| (0,2) | (1,2) | (2,2) | (3,2) | ... |
| (0,3) | (1,3) | (2,3) | (3,3) | ... |
| ⋮     | ⋮     | ⋮     | ⋮     | ⋮   |

→

|       |       |       |   |     |
|-------|-------|-------|---|-----|
|       | $C_2$ | $C_5$ |   | ... |
| $C_1$ | $C_4$ |       |   | ... |
| $C_3$ |       |       |   | ... |
|       |       |       |   | ... |
| ⋮     | ⋮     | ⋮     | ⋮ | ⋮   |

$$\vec{C}(\vec{x}_n) = (\hat{C}_{0,1}, \hat{C}_{1,0}, \hat{C}_{0,2}, \hat{C}_{1,1}, \hat{C}_{2,0})^T.$$

- Из 8 параметров составляется вектор

$$\vec{p}_n = (B(\vec{x}_n), \bar{B}(\vec{x}_n), \sigma(\vec{x}_n), (\vec{C}(\vec{x}_n))^T)^T$$

- Из векторов  $\vec{p}_n$  составляется матрица объект-признак  $P$  (размерности  $K \times N$ ):

$$P = (\vec{p}_1, \vec{p}_2, \dots, \vec{p}_N).$$

- Считаем средний вектор:

$$m = \frac{1}{N} \sum_{n=1}^N \vec{p}_n$$

- Считаем матрицу ковариаций:

$$C = (P - M)^T(P - M), \text{ где } M = \overbrace{(m, m, \dots, m)}^N$$

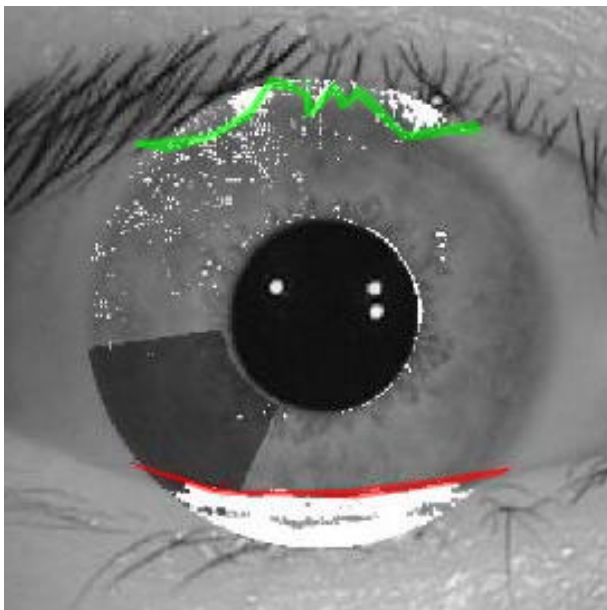
- Получаем оценку вероятности нахождения каждого элемента  $x$  в классе «радужка» через расстояние Махаланобиса:

$$P = \exp \left( (x - m) C^{-1} (x - m)^T \right)$$

- Сравнивая вероятность  $P$  с некоторым пороговым значением  $P_{\text{порог}}$ , относим каждый пиксель к тому или иному классу.

$$\begin{cases} P(x) \geq P_{\text{порог}} \implies x - \text{радужка} \\ P(x) < P_{\text{порог}} \implies x - \text{затенение} \end{cases}$$

# Пример работы алгоритма





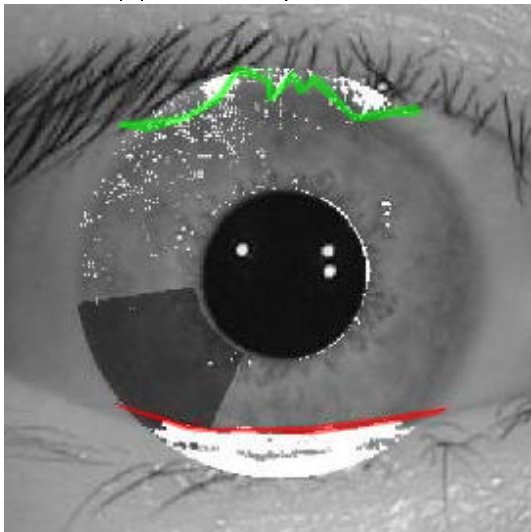
На этом тесте видны две проблемы:

- На изображениях виден **шум**.
- Существует **проблема с распознаванием век** - часть нижнего века сложноотличима от радужки.

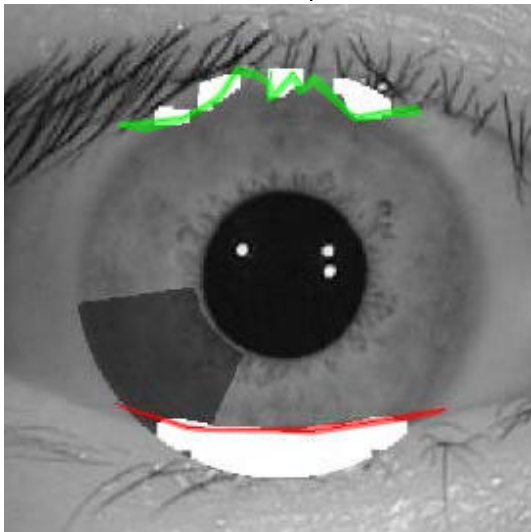
Предлагаемое решение:

- Применить операции математической морфологии - замыкание и размыкание, на логическом изображении вывода алгоритма.
- Для устранения проблемы с нижним веком можно после применения фильтров выделить верхнюю границу нижнего затенения и считать затенением всё, что находится снизу от неё. В данной работе эта проблема не решалась.

До постобработки.

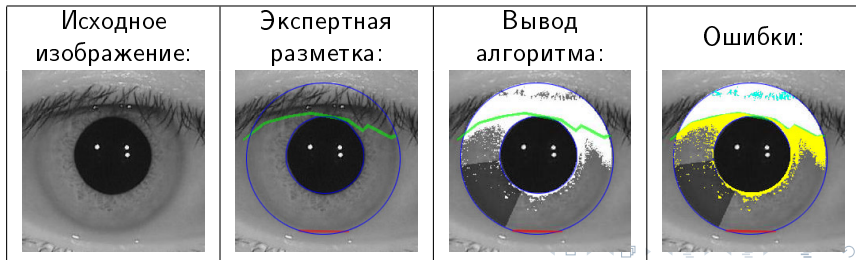


После постобработки.



Введём ф-цию ошибки как сумму относительных ошибок I и II рода. На данном рисунке на изображение последовательно наносится экспертная разметка, затем вывод алгоритма, затем цветом выделяются ошибочно классифицированные пиксели.

- **Жёлтым** цветом выделены пиксели, в которых имеет место ошибка **первого рода**. Т.е. пиксели, ошибочно классифицированные как затенения.
- **Голубым** цветом - ошибка **второго рода**, т.е. пиксели, ошибочно классифицированные как радужка.



Более формально. Пусть  $J$  - вывод,  $\hat{J}$  - экспертная разметка. Пусть:

- $N_0$  - общее количество классифицируемых точек.
- $\hat{N}_0$  - количество точек, классифицируемых как затенения в экспертной разметке.
- $\delta_{\Omega}(i, j)$  - индикаторная функцию кольцевой области  $\Omega$ , аппроксимирующей область радужки.

Т.е.:  $N_0 = |\{(i, j) \mid \delta_{\Omega}(i, j) = 1\}|$ ;  $\hat{N}_0 = |\{(i, j) \mid \delta_{\Omega}(i, j)\hat{J}(i, j) = 1\}|$ .

Тогда относительные ошибки I и II рода выражаются как:

$$E_1 = \frac{1}{\hat{N}_0} \sum_{i=1}^H \sum_{j=1}^W \hat{J}(i, j)(1 - J(i, j))\delta_{\Omega}(i, j),$$

$$E_2 = \frac{1}{N_0 - \hat{N}_0} \sum_{i=1}^H \sum_{j=1}^W J(i, j)(1 - \hat{J}(i, j))\delta_{\Omega}(i, j).$$

Функция ошибки определяется как их сумма:

$$E = E_1 + E_2.$$

При описании метода решения было сформулировано 3 параметра алгоритма:

- $a$  - Сторона квадрата, в котором считается признак  $\overline{B}(\vec{x}_n)$ .
- $b$  - Сторона квадрата, в котором считается признак  $\sigma(\vec{x}_n)$ .
- $P_{\text{порог}}$  - Пороговая вероятность.

При разных параметрах алгоритм работает с разной точностью. Используя для оценки точности распознавания фиксированного изображения  $I$  на фиксированных параметрах описанную выше функцию ошибки  $E(I, a, b, P)$ , оптимизируем метод с помощью генетического алгоритма.

## Генетический алгоритм:

Особь: вектор, состоящий из трёх параметров алгоритма:

$$v = \begin{pmatrix} a \\ b \\ P_{\text{порог}} \end{pmatrix}$$

Функция качества особи:

$$f(v) = f \begin{pmatrix} a \\ b \\ P_{\text{порог}} \end{pmatrix} = \frac{1}{M} \sum_{i=1}^M (1 - E(I_i, a, b, P_{\text{порог}}))$$

Селекция: метод рулетки, т.е. вероятность выбора особи  $v$  в качестве родителя пропорциональна  $f(v)$ .

## Генетические операторы:

$$\text{Скращивание: } v \otimes w = \begin{pmatrix} a_1 \\ b_1 \\ P_1 \end{pmatrix} \otimes \begin{pmatrix} a_2 \\ b_2 \\ P_2 \end{pmatrix} = \begin{pmatrix} r(a_1, a_2) \\ r(b_1, b_2) \\ r(P_1, P_2) \end{pmatrix}.$$

Где  $r(x, y)$  - случайная величина (распределённая по Бернулли):

$$r(x, y) = \begin{cases} x & , \text{ с вероятностью } p = \frac{1}{3} \\ y & , \text{ с вероятностью } p = \frac{1}{3} \\ \frac{x+y}{2} & , \text{ с вероятностью } p = \frac{1}{3} \end{cases}$$



Мутация:

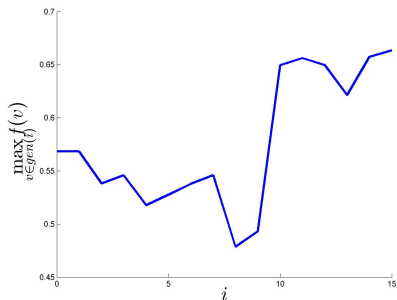
$$M(v) = M \begin{pmatrix} a \\ b \\ P \end{pmatrix} = \begin{cases} v & , \text{ с вероятностью } p = \frac{2}{3} \\ (r_1, b, P)^T & , \text{ с вероятностью } p = \frac{1}{9} \\ (a, r_1, P)^T & , \text{ с вероятностью } p = \frac{1}{9} \\ (a, b, r_2)^T & , \text{ с вероятностью } p = \frac{1}{9} \end{cases}, \text{ где:}$$

$r_1$  - дискретная случ. величина, равномерно распределённая на  $[1, 15]$

$r_2$  - непрерывная случ. величина, равномерно распределённая на  $[0.6, 1]$

# Конечный алгоритм

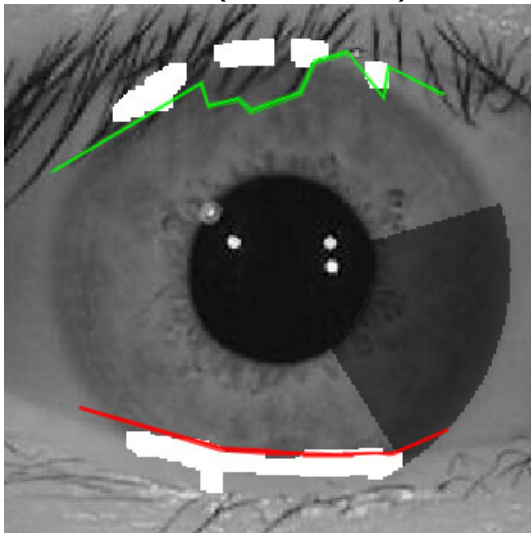
В ходе эксперимента было рассчитано 16 поколений генетического алгоритма для популяции из 10 особей.



Для конечного алгоритма выбран следующий набор параметров:

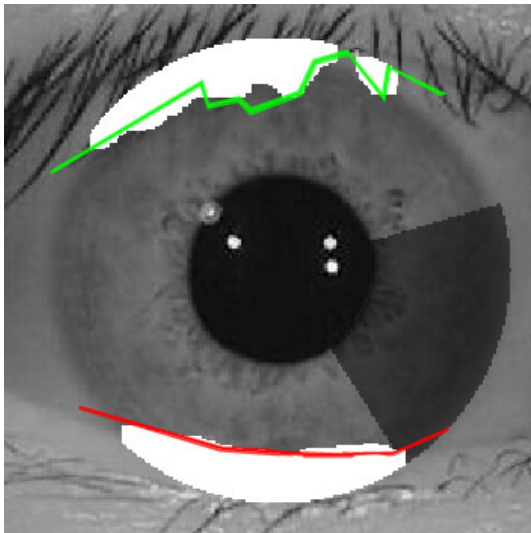
$$v^* = \begin{pmatrix} 10 \\ 7 \\ 0.85 \end{pmatrix}$$

$$v_0 = (5, 5, 0.7)^T$$



# Пример работы на параметрах $v_0$ и $v^*$

$v^*$

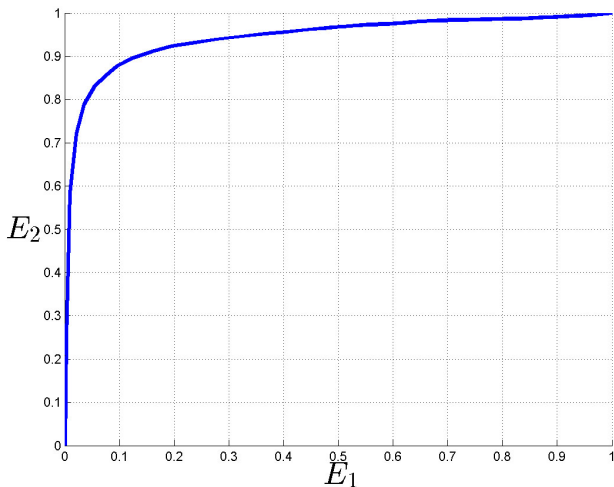


Были протестированы 3 модификации алгоритма:

- 1 Алгоритм с параметрами  $v_0$  без постобработки.
- 2 Алгоритм с параметрами  $v^*$  без постобработки.
- 3 Алгоритм с параметрами  $v^*$  с постобработкой.

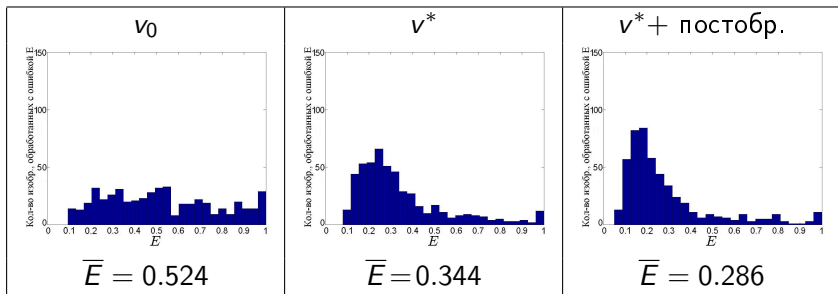
Тестирование проводилось на 500 изображениях из базы CASIA, вычисление ошибки производилось на основе ручной разметки затенений. В качестве функции ошибки использовалась описанная выше функция  $E$ .

ROC-кривая:

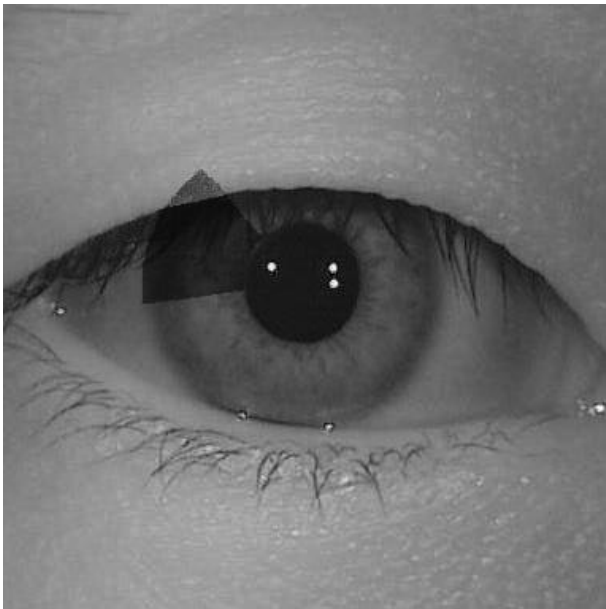


# Точности различных модификаций.

На данных графиках в виде гистограмм представлены распределения функций ошибки каждой из исследуемых модификаций алгоритмов на 500 изображениях базы CASIA:

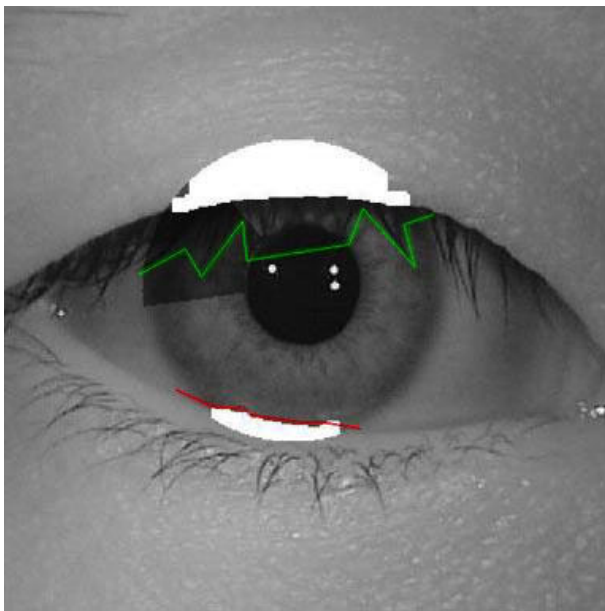


# Проблема с незатенённым сегментом.





# Проблема с незатенённым сегментом.



Представленный метод с достаточно высокой точностью определяет области затенения радужки.

- 1 Основное преимущество метода состоит в том, что он может являться частью **полностью автоматического** метода, не требующего ручной разметки для обучающей выборки.
- 2 Метод строит **новый классификатор для каждого изображения**, что обеспечивает **стабильность** работы на изображениях с разной структурой радужки и с разной освещённостью.
- 3 К сожалению, **скорость работы метода невелика** (среднее время работы на 1 изображении на выборке CASIA составляет 10 секунд), однако сейчас решение переписывается на C++, что, возможно, повлечёт уменьшение времени работы.

Спасибо за внимание!