LOBACHEVSKY UNIVERSITY

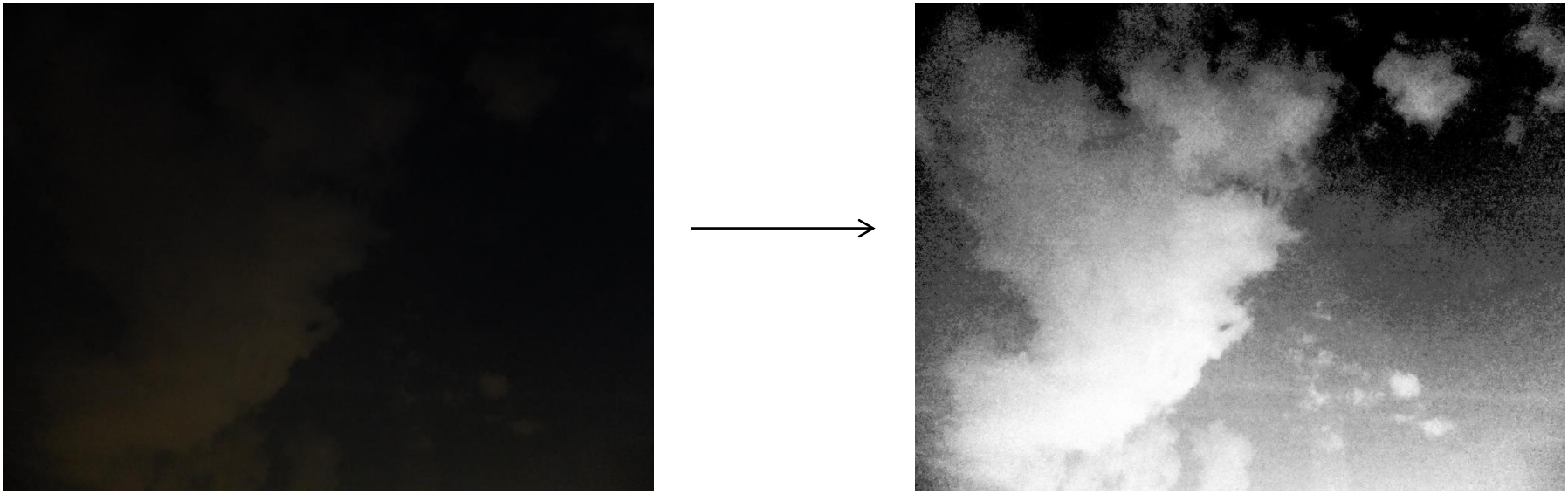
itseez

Обработка изображений

Сиднев Алексей

05.07.2016

- Улучшение изображений:
  - Визуальное восприятие
  - Повышение качества дальнейшей обработки



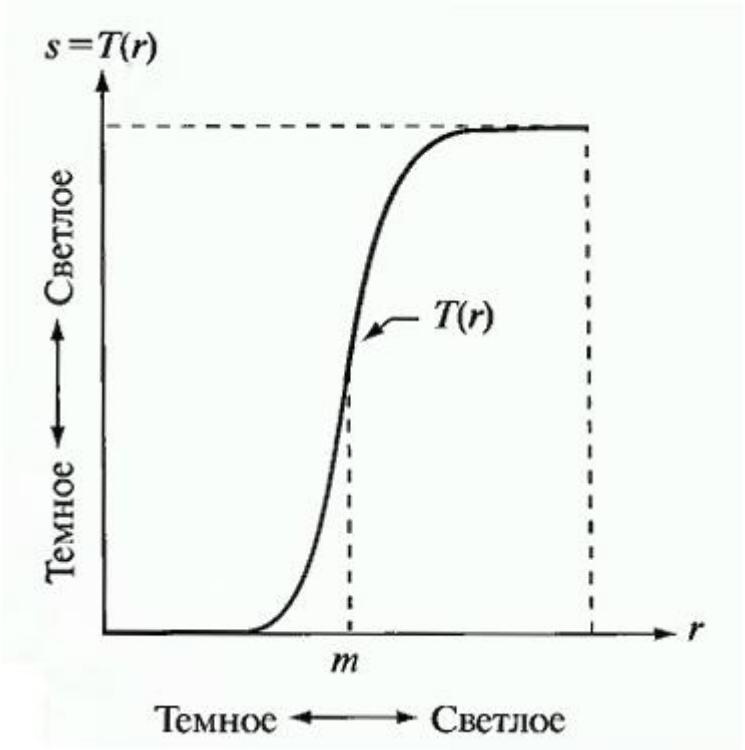
LOBACHEVSKY UNIVERSITY itseez

$$f(x,y) = T[g(x,y)],\ g(x,y) \in [0, L-1]$$

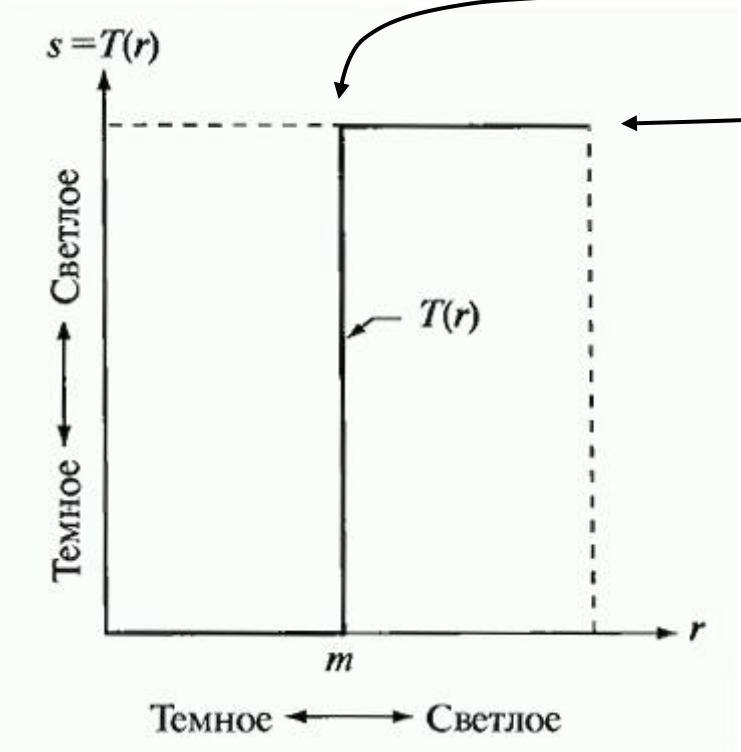
T – оператор, определённый в окрестности (x,y)

- Градационное преобразование (окрестность 1x1)
- Выравнивание гистограммы
- Арифметико-логические операции
- Фильтрация по области
  - Размытие
  - Медианный фильтр
  - Повышение резкости

s = T(r), r – исходная яркость, s – результирующая яркость



```
double threshold(InputArray src, OutputArray dst,
                 double thresh, double maxval, int type)
```



type = THRESH_BINARY

Негатив

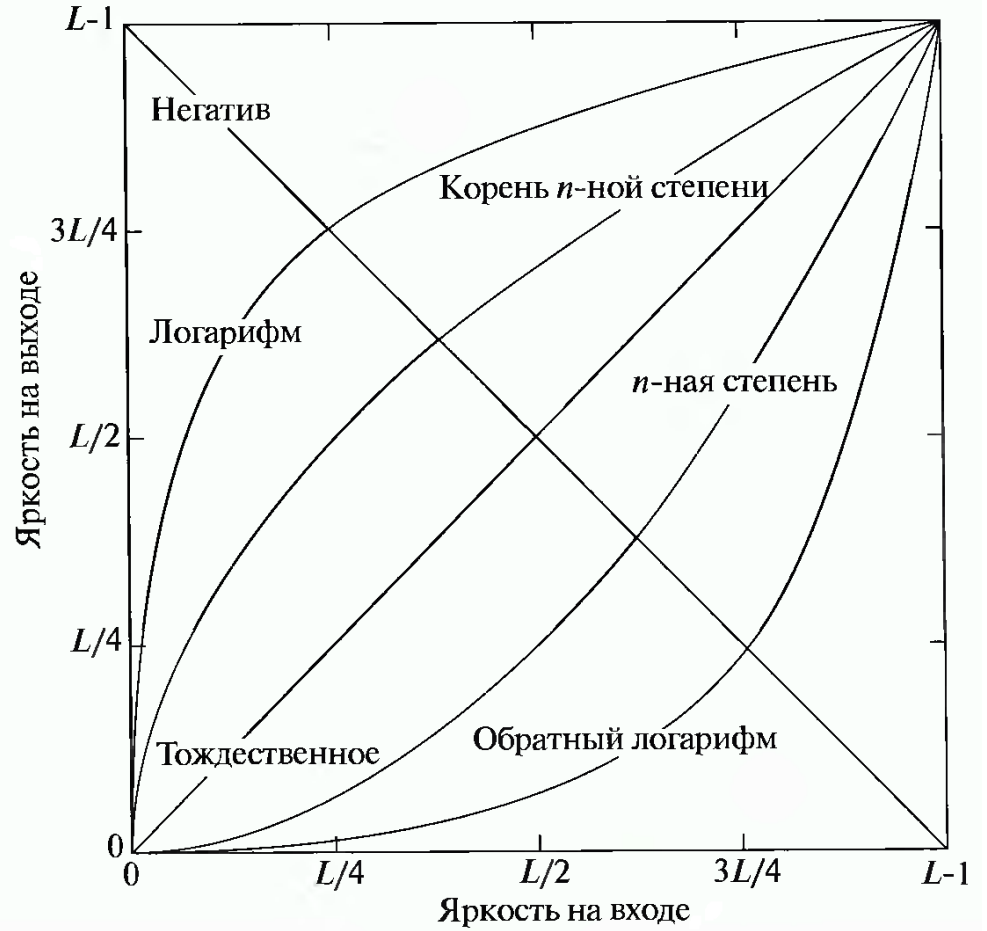
$$s = L - 1 - r$$

Логарифмическое преобразование

$$s = c \cdot log(1 + r)$$

Степенное преобразование (гамма-коррекция)

$$s = c \cdot r^{\gamma}$$

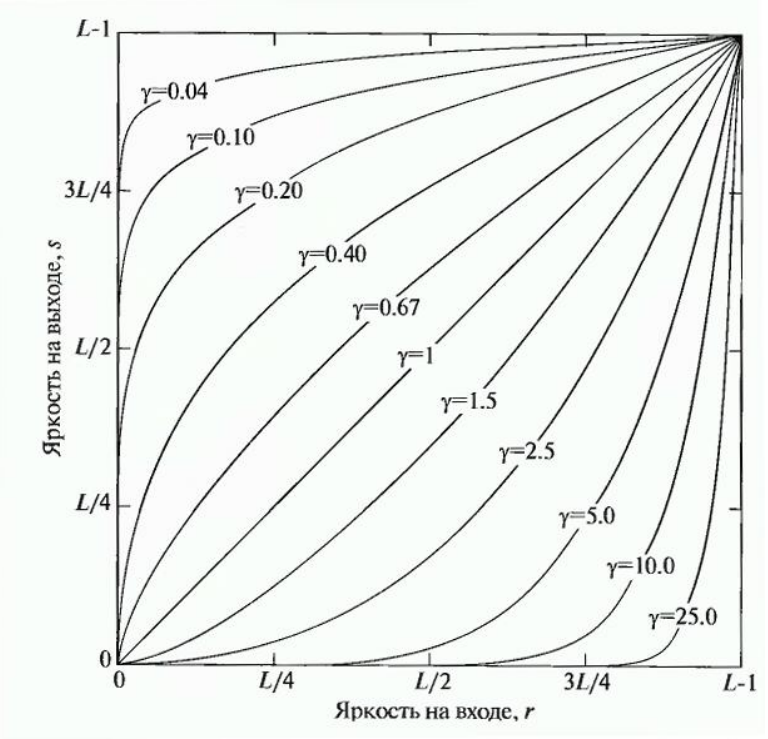


LOBACHEVSKY UNIVERSITY itseez

```
Mat CorrectGamma(Mat& img, double gamma) {
    Mat lut_matrix(1, 256, CV_8UC1);
    uchar * ptr = lut_matrix.ptr();
    for (int i = 0; i < 256; i++)
        ptr[i] = pow(i / 255.0, gamma) * 255.0;

    Mat result;
    LUT(img, lut_matrix, result);

    return result;
}
```



LOBACHEVSKY UNIVERSITY itseez

- Гистограмма

$$h(r_k) = n_k, k \in [0, L-1]$$

r_k – k-ый уровень яркости

n_k – число пикселей на с яркостью r_k

- Нормализованная гистограмма

$$p(r_k) = n_k/n, \; n = \sum n_k$$

```
void calcHist(const Mat * images, int nimages, const int * channels,
              InputArray mask, OutputArray hist, int dims,
              const int * histSize, const float ** ranges,
              bool uniform = true, bool accumulate = false)
```

$$s = T(r)$$

$p_r(r)$
$p_s(s)$ – плотности распределения

$$p_s(s) = p_r(r) \cdot \frac{dr}{ds}$$

$$s = T(r) = \int_0^r p_r(w)dw \quad \Rightarrow \quad p_s(s) = 1$$

$$p_r(r_k) = n_k/n \qquad n = \sum n_k \quad k \in [0, L-1]$$

$$s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} \frac{n_j}{n}$$

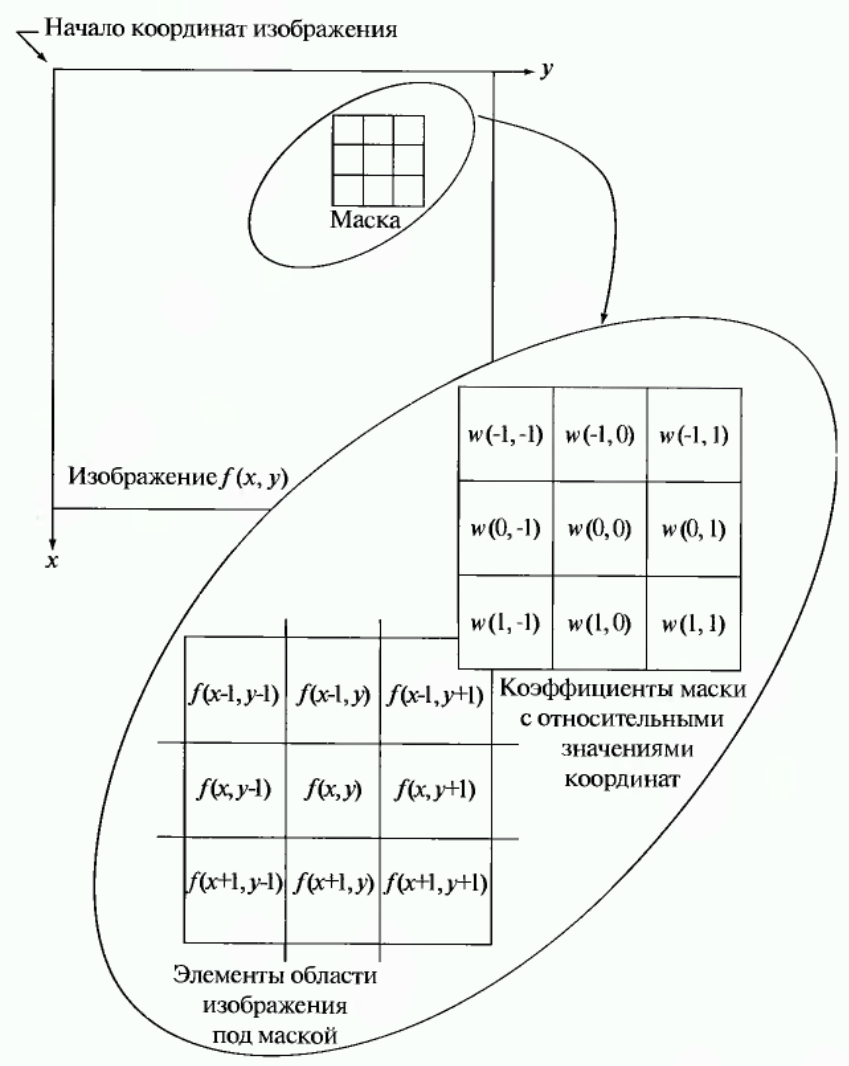
```
void equalizeHist(InputArray src, OutputArray dst)
```

- NOT – негатив
- OR, AND – маскирование
- Вычитание
  - Контрастное вещество
  - Выделение границ
- Сложение
  - Усреднение изображений – подавление шума

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) \cdot f(x+s, y+t)$$

Границы:

- Не обрабатывать
- Дополнять граничными пикселями
- Применять маску к имеющимся пикселям



LOBACHEVSKY UNIVERSITY itseez

Сумма элементов

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

```
void boxFilter(InputArray src, OutputArray dst,
               int ddepth, Size ksize,
               Point anchor = Point(-1, -1),
               bool normalize = true,
               int borderType = BORDER_DEFAULT)
```

Взвешенная сумма

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

```
void filter2D(InputArray src, OutputArray dst,
              int ddepth, InputArray kernel,
              Point anchor = Point(-1, -1),
              double delta = 0,
              int borderType = BORDER_DEFAULT)
```

- Медианный фильтр
  - Выбор среднего из окрестности ksize x ksize

```
void medianBlur(InputArray src, OutputArray dst, int ksize)
```

- Оператор Собеля

-1	-2	-1
0	0	0
1	2	1

$$\nabla f = ((\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2)^{1/2} \approx |\frac{\partial f}{\partial x}| + |\frac{\partial f}{\partial y}|$$

-1	0	1
-2	0	2
-1	0	1

```
void Sobel(InputArray src, OutputArray dst, int ddepth,
    int dx, int dy, int ksize = 3, double scale = 1,
    double delta = 0, int borderType = BORDER_DEFAULT)
```

- Лапласиан

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

0	1	0
1	-4	1
0	1	0

Повышение резкости

$$g(x,y) = f(x,y) - \nabla^2 f(x,y)$$

```
void Laplacian(InputArray src, OutputArray dst, int ddepth,
               int ksize = 1, double scale = 1, double delta = 0,
               int borderType = BORDER_DEFAULT)
```

Гауссово размытие

LOBACHEVSKY UNIVERSITY itseez

```
void GaussianBlur(InputArray src, OutputArray dst,
                  Size ksize, double sigmaX, double sigmaY = 0,
                  int borderType = BORDER_DEFAULT)
```

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

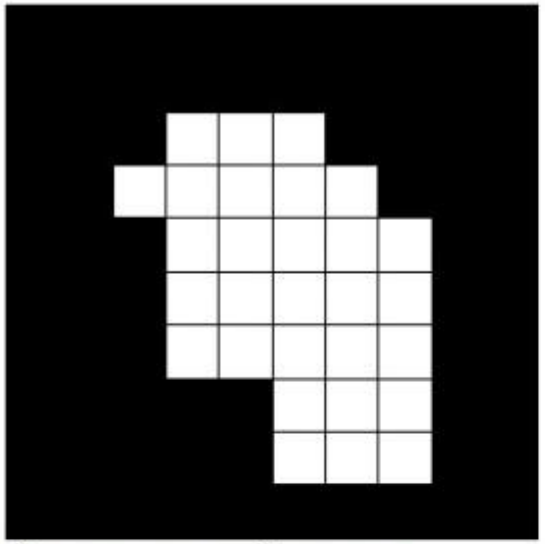
$$G_i = \alpha * e^{-(i-(\mathtt{ksize}-1)/2)^2/(2*\mathtt{sigma}^2)}$$

- Дилатация
  - Локальный максимум по шаблону

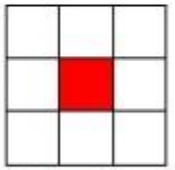
```
void dilate(InputArray src,
   OutputArray dst,
   InputArray kernel,…)
```

- Эрозия
  - Локальный минимум по шаблону

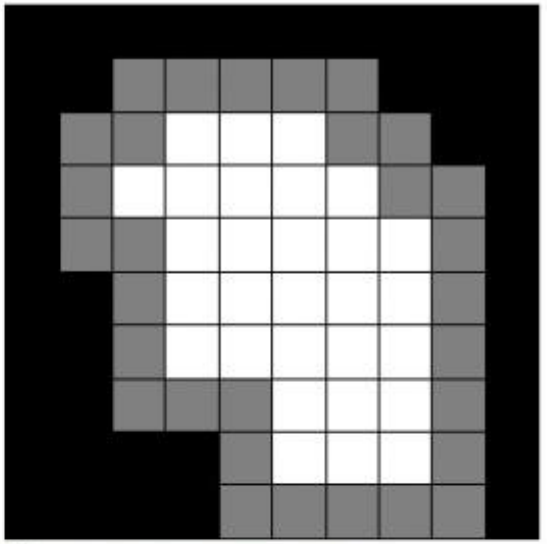
```
void erode(InputArray src,
   OutputArray dst,
   InputArray kernel,…)
```



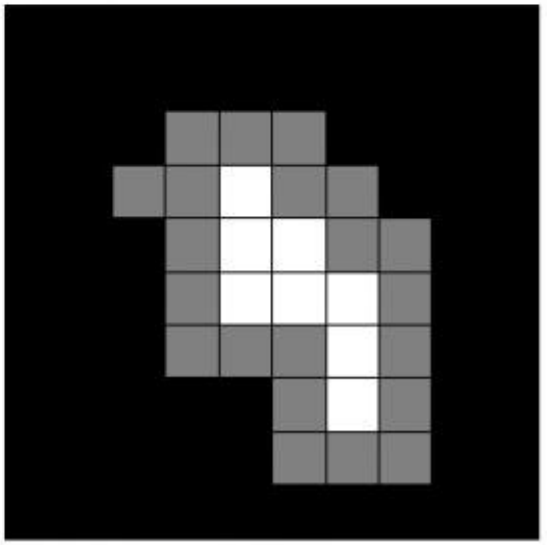
a) исходное изображение



b) шаблон (центр – ведущий элемент)



c) результат дилатации



d) результат эрозии

```
void morphologyEx(InputArray src, OutputArray dst,
    int op, InputArray kernel, Point anchor = Point(-1, -1),
    int iterations = 1, int borderType = BORDER_CONSTANT,
    const Scalar & borderValue = morphologyDefaultBorderValue())
```

- Типы морфологической операции:
  - MORPH_OPEN – размыкание, удаление шума, dst = open(src, element) = dilate(erode(src, element)).
  - MORPH_CLOSE – замыкание, удаление “дырок”, dst = close(src, element) = erode(dilate(src, element)).
  - MORPH_GRADIENT – морфологический градиент (поиск контуров), dst = dilate(src, element) - erode(src, element).
  - MORPH_TOPHAT – "верх шляпы" ("top hat"), выделение ярких областей, dst = src – open(src, element).
  - MORPH_BLACKHAT – "черная шляпа" ("black hat"), выделение темных областей, dst = close(src, element)- src.

```
void Canny(InputArray image, OutputArray edges,
    double threshold1, double threshold2,
    int apertureSize = 3, bool L2gradient = false)
```

1. Фильтр Гаусса
2. Поиск градиентов (оператор Собеля)
   - Вычисление $atan2(\frac{\partial f}{\partial y}, \frac{\partial f}{\partial x})$
   - Округление угла: 0, 45, 90, 135
3. Подавление немаксимумов (утончение границы)
4. Двойная пороговая фильтрация
5. Трассировка области неоднозначности

LOBACHEVSKY UNIVERSITY itseez

Спасибо за внимание!

e-mail: alexey.sidnev@gmail.com