Лабораторные работы по курсу

Моделирование ч.3 «Компьютерное зрение»

Лабораторная работа 2

Бинаризация изображений

2021

Содержание

Введение

Целью лабораторной работы является освоение навыков программной реализации адаптивных алгоритмов бинаризации изображений и исследование качества их работы в сравнении с оптимальными алгоритмами.

Лабораторная работа содержит два задания для самостоятельного выполнения. Первое задание состоит из реализации и исследования алгоритма адаптивной бинаризации и анализ его характеристик. Второе задание заключается в реализации дополнительных метрик оценки качества работы алгоритмов бинаризации.

Лабораторная работа рассчитана на 4 академических часа.

1. Краткая теоретическая справка

1.1. Бинаризация изображений

В компьютерном зрении бинаризация изображений часто применяется в задачах распознавания текста, картографии, анализе медицинских снимков, поиска дефектов при контроле качества изделий и т. д. Процесс бинаризации представляет собой перевод полутонового изображения в двухцветное черно-белое (Рисунок Рисунок 1).

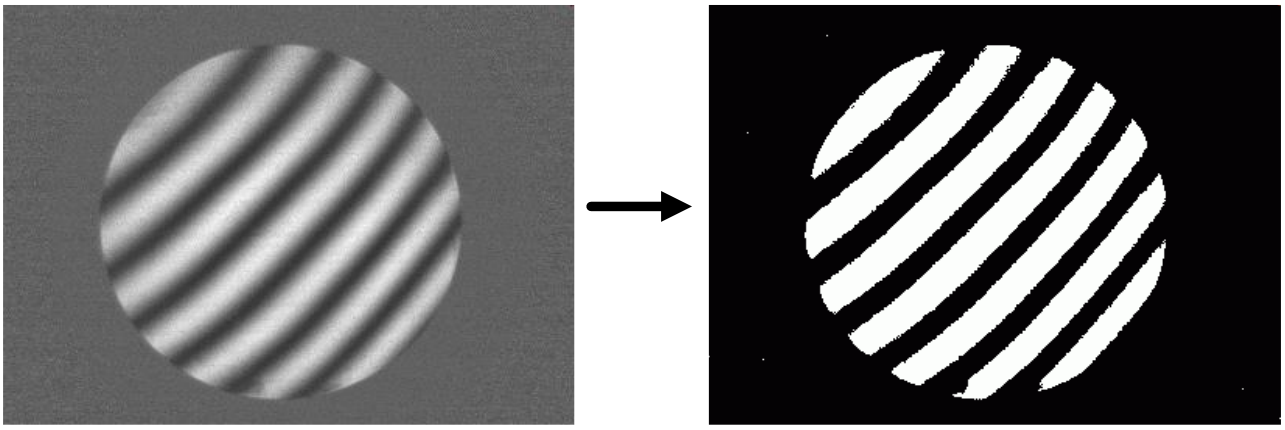

Рисунок 1. Бинаризация изображения

Как правило, все отсчеты изображения, относящиеся к фону, маркируются одним цветом, а все отсчеты изображения, составляющие полезный объект (сигнал), маркируются другим цветом. В результате в алгоритмах бинаризации необходимо принять решение относительного каждого отсчета изображения о его принадлежности либо к фону, либо к сигналу. Ввиду вероятностной природы уровней интенсивности в отсчетах, решение принимается на основе сравнения значения отсчета изображения с некоторым пороговым значением:

$$I'(x,y) = \begin{cases} 0, \text{ если } I(x,y) > h \\ 1, \text{ если } I(x,y) \le h \end{cases},$$

где $I(x,y)$ – входное полутоновое изображение, $I'(x,y)$ – выходное бинарное изображение, h – порог.

При бинаризации каждого отсчета могут быть допущены ошибки. Ситуацию, при которой в отсчете изображения не было сигнала (гипотеза H_0), но он был маркирован при бинаризации как сигнальный (гипотеза H_1), называют ошибкой первого рода или ложным срабатыванием. Ситуацию, при которой в отсчете изображения был сигнал (гипотеза H_1), но он был маркирован при бинаризации как фоновый (гипотеза H_0), называют ошибкой второго рода или пропуском сигнала.

Наибольшей сложностью в процедуре бинаризации является вычисление порога таким образом, чтобы обеспечить минимизацию ошибок.

1.2. Оптимальное обнаружение сигнала

Процесс бинаризации изображения может быть рассмотрен с точки зрения задачи обнаружения сигнала. Для сигнала на фоне белого аддитивного гауссовского шума оптимальным решающим правилом, базирующемся на отношении правдоподобия, является

$$q = \sum_{y=1}^{V} \sum_{x=1}^{H} I(x,y) \cdot s(x,y) > h,$$

где D – дисперсия шума, s – функция, описывающая полезный сигнал. Порог h вычисляется исходя из нескольких критериев. Одним из наиболее часто применяемых критериев является критерий идеального наблюдателя:

$$h = \frac{E}{2} + D \cdot \ln\left(\frac{p_{pr}(H_0)}{p_{pr}(H_1)} \right),$$

где E – энергия сигнала:

$$E = \sum_{(x,y) \in S} s(x,y)^2,$$

$p_{pr}(H_0)$ и $p_{pr}(H_1)$ – априорные вероятности отсутствия (гипотеза H_0) или присутствия (гипотеза H_1) полезного сигнала в отсчете изображения. В случае отсутствия априорной информации порог сравнения может быть вычислен только на основе информации об энергии сигнала:

$$h = \frac{E}{2}.$$

Порог, рассчитанный по критерию идеального наблюдателя, обеспечивает минимизацию ошибок первого и второго рода.

Еще одним часто применяемым критерием является критерий Неймана-Пирсона, позволяющий зафиксировать количество ошибок первого рода. В инженерной практике порог обнаружения рассчитывается, как правило, по следующей формуле:

$$h = \mu + k \cdot \sigma,$$

где μ – математическое ожидание фона, σ – среднеквадратичное отклонение шума, k – константа, обычно равная 3.

Для применения оптимальных методов расчета порога h необходимо знание энергии сигнала и/или параметров распределения шума. В задачах бинаризации изображений зачастую данные величины и не могут быть корректно оценены. В результате, применяемые на практике алгоритмы бинаризации в первую очередь нацелены на прямую или косвенную оценку данных величин и расчета порога сравнения h исходя из полученных оценок.

1.3. Метод бинаризации Оцу

Метод бинаризации Оцу был опубликован в работе [1]. Данный метод является методом глобального порога, то есть значение порога рассчитывается по всей площади изображения.

Основой метода является представление изображения как бимодального распределения, где один пик соответствует фону, а второй пик – сигналу. Оптимальным положением порога считается такое разделение отсчетов изображения на сигнальные и фоновые, при котором минимизируется внутригрупповая дисперсия.

Метод состоит из следующих шагов:

1. по всем отсчетам изображения строится гистограмма;

2. значение порога последовательно устанавливается от минимального до максимального значения отсчета;

3. для каждого положения порога рассчитываются дисперсии среди отсчетов меньше порога и среди отсчетов больше порога:

$$D_{0i} = DISP(0, h_i)$$
$$D_{1i} = DISP(h_i + 1, \mathrm{I}_{max})$$

4. для полученных значений дисперсий рассчитывается критерий разделимости на группы

$$SC(h_i) = 1 - \frac{D_{0i} - D_{1i}}{DISP(0, \mathrm{I}_{max})}$$

5. выбирается то значение порога, при котором максимизируется дисперсия между группами:

$$h = \underset{h_i \in \{0,\ldots,\mathrm{I}_{max}\}}{\arg\max} \left(SC(h_i) \right)$$

Порог, рассчитываемый методом Оцу, по смыслу близок к порогу, получаемому по критерию идеального наблюдателя. Недостатком метода Оцу являются

предположения о бимодальности распределения и предположение о стационарности параметров фона и сигнала на всей плоскости изображения.

1.4. Адаптивные алгоритмы бинаризации изображений

Адаптивные алгоритмы бинаризации вычисляют значение порога в локальной окрестности отсчета для которого в текущий момент времени принимается решение о маркировании сигналом или фоном.

Для реализации процедуры бинаризации по изображению перемещается окно, как правило, прямоугольной (квадратной) формы. Размеры окна выбираются исходя из предположения о стационарности параметров фона и сигнала внутри локального окна. Для каждого отсчета, находящегося в центре окна, вычисляется свое пороговое значение на основе остальных отсчетов внутри окна.

В настоящее время существует множество различных алгоритмов адаптивной бинаризации.

1.4.1. Алгоритм Бернсена

Алгоритм адаптивной бинаризации Бернсена был опубликован в [2]. Он состоит из следующих шагов.

1. Внутри окна вычисляются максимальный $\mathrm{I}_{\max}$ и минимальный $\mathrm{I}_{\min}$ отсчеты.

2. Порог вычисляется как среднее арифметическое минимального и максимального отсчетов:

$$h = \frac{\mathrm{I}_{\max} + \mathrm{I}_{\min}}{2}$$

3. Если порог превышает заданную константу контраста ε, то для бинаризации центрального отсчета в окне применяется следующее решающее правило:

$$I_{bin} = I > h$$

4. Если порог не превышает ε, то применяется решающее правило сравнения центрального пикселя с фиксированным ранее заданным порогом:

$$I_{bin} = I > t_h$$

Константы ε и t_h выбираются эмпирически. Как правило, $\varepsilon \approx 15$, а $t_h = 127$ для 256-цветного изображения.

1.4.2. Алгоритм Ниблэка

Алгоритм Ниблэка был опубликован в [3]. Он состоит из следующих шагов.

1. Внутри локального окна рассчитывается математическое ожидание

$$\mu = \frac{1}{N} \cdot \sum_{I \in W} I$$

и среднеквадратичное отклонение

$$\sigma = \sqrt{\frac{1}{N} \cdot \sum_{I \in W} (I - \mu)},$$

где W – локальное окно, N – количество отсчетов в локальном окне, I – все отсчеты, находящиеся в локальном окне.

2. Порог в локальном окне рассчитывается как

$$h = \mu + k \cdot \sigma$$

3. Вычисленное значение порога используется в решающем правиле для бинаризации центрального отсчета в локальном окне:

$$I_{bin} = I > h$$

Значение параметра k устанавливается эмпирически из диапазона, как правило, от 0 до 3.

1.4.3. Алгоритм Саувола

Алгоритм Саувола был опубликован в работе [4]. Он состоит из следующих шагов.

1. Внутри локального окна рассчитывается математическое ожидание

$$\mu = \frac{1}{N} \cdot \sum_{I \in W} I$$

и среднеквадратичное отклонение

$$\sigma = \sqrt{\frac{1}{N} \cdot \sum_{I \in W} (I - \mu)},$$

где W – локальное окно, N – количество отсчетов в локальном окне, I – все отсчеты, находящиеся в локальном окне.

2. Порог в локальном окне рассчитывается как

$$h = \mu \cdot \left(1 + k \cdot \left(\frac{\sigma}{R} - 1\right)\right)$$

3. Вычисленное значение порога используется в решающем правиле для бинаризации центрального отсчета в локальном окне:

$$I_{bin} = I > h$$

Значения параметров k и R подбираются эмпирически. Как правило, k выбирается из диапазона от 0,2 до 0,5. Параметр R составляет 128.

1.5. Метрики качества работы алгоритмов бинаризации

Для оценки качества работы алгоритмов бинаризации применяются различные метрики. Как правило, их можно разделить на два класса:

- метрики, использующие сравнение с эталонным изображением
- метрики, не использующие сравнение с эталонным изображением

1.5.1. С применением эталонного изображения

Метрики сравнения результирующего бинаризированного изображения с эталонным позволяют получить прямые оценки для ошибок первого и второго рода.

Ошибка первого рода (ложное срабатывание), выраженная в процентах, может быть измерена как отношение количества ложно определенных отсчетов как сигнальные к общему количеству фоновых отсчетов на эталонном изображении:

$$err_1 = \frac{\left|I'_{sig}\right| - \left|I_{sig} \cap I'_{sig}\right|}{TotalPix - \left|I_{sig}\right|} \cdot 100\%$$

где I_{sig} – отсчеты эталонного изображения, являющиеся сигнальными, I'_{sig} – отсчеты результирующего бинарного изображения, отмеченные как сигнальные, $TotalPix$ – общее количество отсчетов изображения.

Ошибка второго рода (пропуск сигнала), выраженное в процентах, может быть измерено как отношение количество неправильно определенных отсчетов как фоновые к общему количеству сигнальных отсчетов на эталонном изображении:

$$err_2 = \frac{\left|I_{sig}\right| - \left|I_{sig} \cap I'_{sig}\right|}{\left|I_{sig}\right|} \cdot 100\%$$

1.5.2. Без применения эталонного изображения

Без использования эталонного изображения применяются статистические метрики бинаризации, количественно выражающие субъективное восприятие качества бинаризации.

Оценка однородности бинаризации может быть произведена через вычисление дисперсий:

$$U = 1 - \frac{A_S \cdot \sigma_S^2}{TotalPix \cdot \sigma^2},$$

где σ_S^2 – дисперсия интенсивности отсчетов, которые были маркированы при бинаризации как сигнальные, σ^2 – дисперсия всех отсчетов исходного изображения.

Оценка контраста между фоновыми и сигнальными отсчетами может быть дана как

$$C = \frac{|\mu_S - \mu_B|}{\mu_S + \mu_B},$$

где μ_S – среднее арифметическое отсчетов, маркированных сигнальными, μ_B – среднее арифметическое отсчетов, маркированных как фоновые.

2. Программные инструменты

2.1. Описание проекта *Binarizing* для *PyCharm IDE*

Проект *Binarizing* представляет собой написанный на языке *python 3* [5] исходный код в среде PyCharm IDE для исследования работы алгоритмов бинаризации в зависимости от отношения сигнал-шум.

Для работы исходного кода используются следующие библиотеки для языка *python*:

- *OpenCV (opencv-python)*
- *NumPy*
- *Matplotlib*

Проект состоит из следующих файлов исходных кодов:

- *main.py*
- *decetrors.py*
- *metrics.py*
- *show.py*

Файл *main.py* является основным исполняемым файлом проекта и содержит в себе процедуру исследования алгоритмов бинаризации. На Рисунке Рисунок 2 представлена блок-схема процедуры исследования.

Вначале производится инициализация параметров, констант и фигуры для отображения промежуточных результатов. Затем на каждой итерации цикла формируется тестовое изображение с фиксированным отношением сигнал-шум и обрабатывается различными алгоритмами бинаризации. Затем для каждого алгоритма измеряются значения ошибок первого и второго рода. После завершения цикла на экран выводятся графики зависимостей ошибок от отношения сигнал-шум для каждого алгоритма бинаризации.

Файл *detectors.py* содержит в себе функции, реализующие различные алгоритмы бинаризации тестового изображения.

Файл *metrics.py* содержит в себе функции для вычисления метрик качества работы алгоритмов бинаризации.

Файл *show.py* содержит в себе вспомогательные функции для отображения данных.

В процессе исследования алгоритмов бинаризации промежуточный результат обработки выводится на экран (Рисунок Рисунок 3).

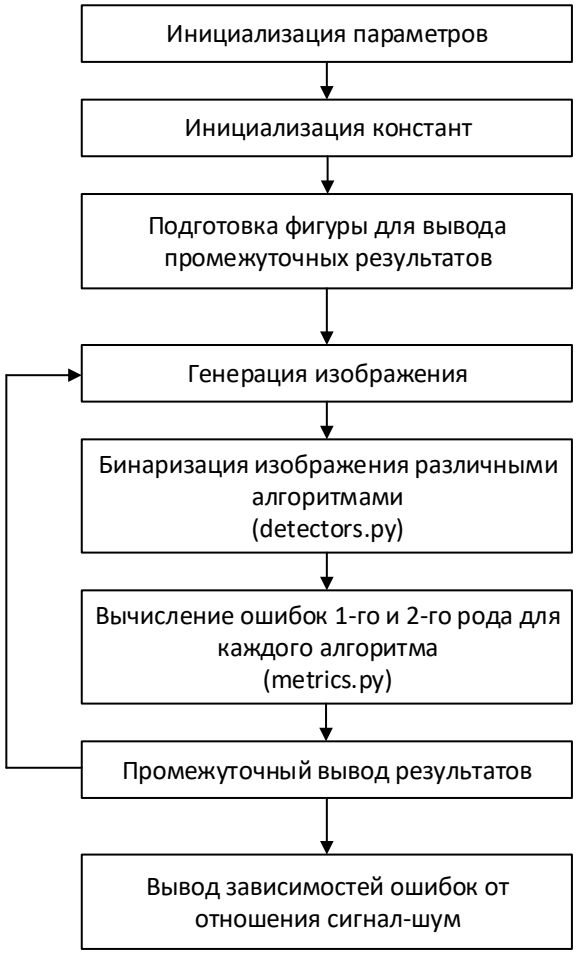

Рисунок 2. Блок-схема процедуры исследования алгоритмов

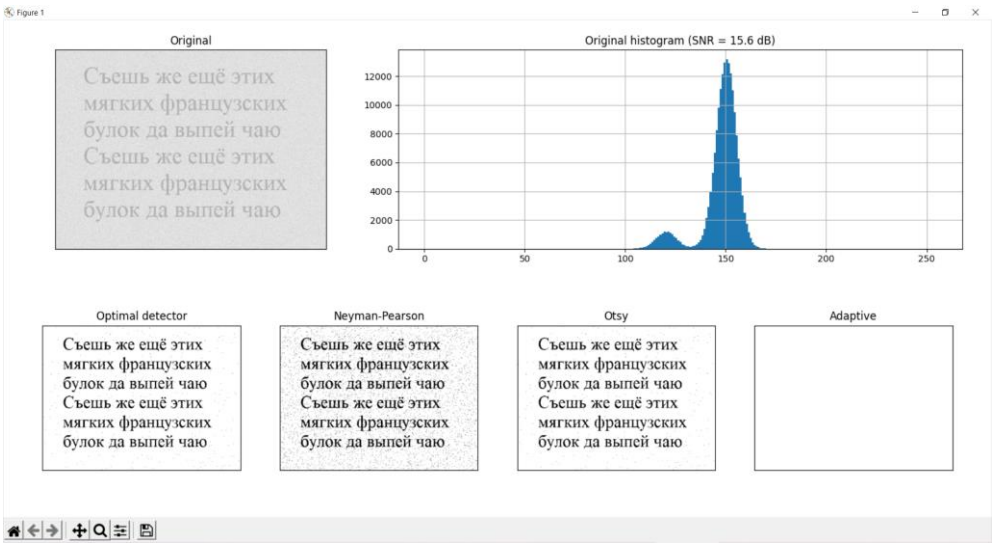

Рисунок 3. Вывод промежуточных данных

После завершения обработки тестовых изображений зависимости ошибок от отношения сигнал-шум выводятся на экран (Рисунок Рисунок 4).

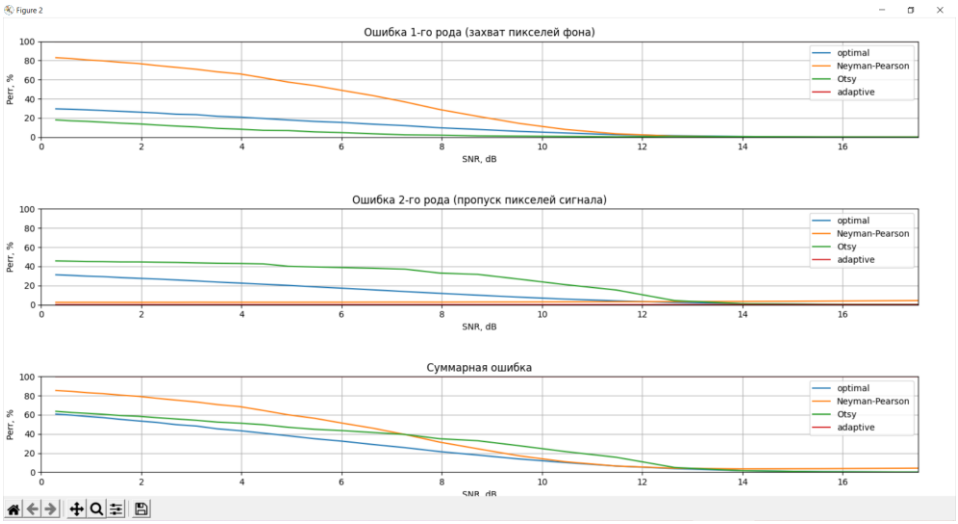

Рисунок 4. Вывод зависимости ошибок бинаризации от отношения сигнал-шум

Итоговые графики содержат для всех исследуемых алгоритмов количество ошибок первого рода в зависимости от отношения сигнал-шум, количество ошибок второго рода в зависимости от отношения сигнал-шум и суммарное количество ошибок в зависимости от отношения сигнал-шум.

2.2. Генерация тестовых изображений

Тестовое изображение для исследования алгоритмов бинаризации представляет собой модель зашумленного текста. Оно генерируется в исходном коде *main.py* (комментарий «# 3.1.») на основе эталонного изображения текста с низкой контрастностью и белого аддитивного гауссовского шума.

Путь к эталонному изображению указывается в переменной TEST_IMG_FILE. По умолчанию он равен «../data/tst_img.png». На Рисунке Рисунок 5 представлено эталонное изображение.

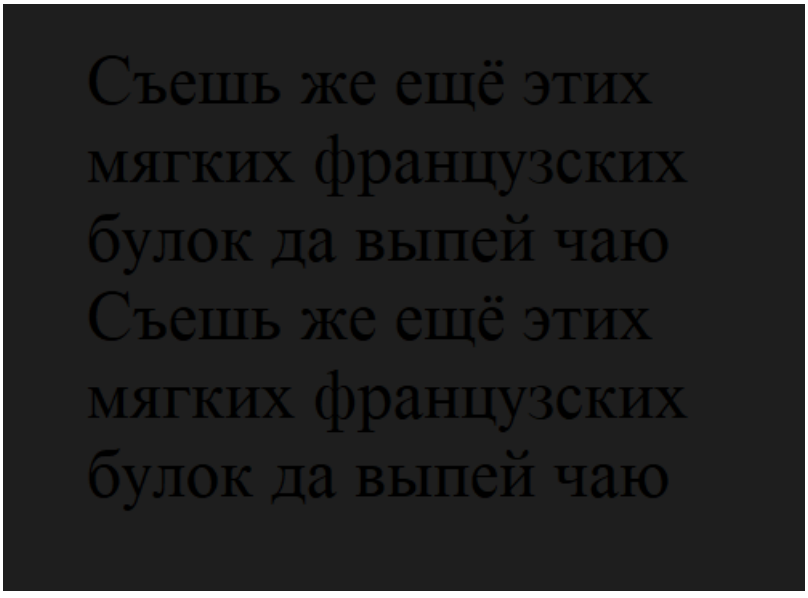

Рисунок 5. Эталонное изображение для исследования алгоритмов

Различные отношения сигнал-шум моделируются с помощью изменения среднеквадратичного отклонения шума. Диапазон среднеквадратичных отклонений задается в переменной «*sigmas*». Для каждого фиксированного значения среднеквадратичного шума **σ** из *sigmas* отношение сигнал-шум рассчитывается как

$$\mathrm{SNR} = 10\log_{10}(\frac{s^2}{\sigma^2})\ \text{дБ}$$

За сигнал *s* принимается уровень яркости в отчете идеального изображения с координатами (0, 0). Математическое ожидание шума задается в переменной «*mu*».

2.3. Бинаризация изображений

В файле *detectors.py* реализованы следующие алгоритмы бинаризации:

- функция «*optimal*» – бинаризация оптимальным методом по критерию идеального наблюдателя,
- функция «*Neyman-Pearson*» – бинаризация с помощью критерия Неймана-Пирсона,
- функция «*Otsy*» – алгоритм бинаризации Оцу,
- функцию «*adaptive*» –заготовка для реализации адаптивного алгоритма бинаризации.

Функции «*optimal*» и «*Neyman-Pearson*» используются известные параметры шума и сигнала, которые задаются для генерации тестового изображения.

Функция *«Otsy*» производит оценку параметров изображения для бинаризации в соответствии с алгоритмом Оцу, реализованном в библиотеке *OpenCV*.

Функция «*adaptive»* предназначена для выполнения индивидуального задания по реализации адаптивного алгоритма бинаризации. В ней реализована программная логика выбора отсчетов изображения с помощью локального окна, цикл прохода по набору локальных окон для расчета локальных пороговых значений и решающее правило сравнения входного изображения с пороговым значениями.

2.4. Измерение качества работы алгоритмов бинаризации

В файле *metrics.py* реализованы следующие функции для оценки качества работы алгоритмов бинаризации:

- функция «*I_error*» – функция для измерения количества ошибок первого рода (ложный захват отсчетов изображения);
- функция «*II_error*» – функция для измерения количества ошибок первого рода (пропуск сигнальных отсчетов);

Ошибки первого и второго рода вычисляются с помощью сравнения идеального изображения (*ideal_img*) и изображения, полученного после бинаризации (*bin_img*).

3. Подготовка к выполнению лабораторной работы

3.1. Изучить теоретический материал, представленный в лабораторном практикуме.

3.2. Изучить дополнительную литературу.
3.3. Изучить исходный код проекта *Binarizing* для *PyCharm IDE*
3.4. Изучить работу оптимального детектора, детектора с критерием Неймана-Пирсона и детектора на основе метода Оцу.
3.5. Запустить проект *Binarizing*.
3.6. Проанализировать графики ошибок для разных алгоритмов бинаризации в зависимости от отношения сигнал-шум.

4. Индивидуальные задания

4.1. Задание №1

4.1.1. Реализовать алгоритм адаптивной бинаризации в функции «*adaptive*» файла «*detectors.py*». Алгоритм выбирается в зависимости от

$$i = (N \ \% \ 3),$$

где N – ваш номер в общем списке группы.

- $i = 0$: алгоритм Бернсена
- $i = 1$: алгоритм Ниблэка
- $i = 2$: алгоритм Саувола

4.1.2. Эмпирическим путем подобрать оптимальные значения настроечных параметров алгоритма для минимизации суммарной ошибки бинаризации.

4.2. Задание №2

4.2.1. Доработать исходный код проекта «*Binarizing*», добавив в файл «*metrics.py*» следующие метрики качества работы алгоритмов бинаризации:

- $i = 0$: метрику однородности сегментов
- $i = 1$: метрику контраста между сегментами

4.2.2. Провести измерение добавленной метрики для каждого алгоритма бинаризации в зависимости от отношения сигнал-шум.
4.2.3. Вывести в отдельном окне результирующие графики зависимости для каждого алгоритма и проанализировать результат.

5. Список литературы

1. Otsu N. A threshold selection method from gray-level histograms // IEEE Transactions on Systems, Man, and Cybernetics , Vol. 9, No. 1, Jan. 1979. pp. 62 - 66.
2. Bernsen J. Dynamic thresholding of grey-level images, 1986. pp. 1251-1255.
3. Niblack W. An introduction to Digital Image Processing // Prentice-Hall, 1986.

4. Pietikainen J.S.A.M. Adaptive document image binarization // Pattern Recognition, 2000. pp. 225-236.
5. Python [Электронный ресурс] // Python: [сайт]. URL: https://www.python.org/