

Яркостные преобразования

Лекция 2

Почему это важно?

Примеры из жизни:

- улучшение фотографий в смартфоне (фильтры)
- повышение контрастности медицинских снимков (рентген, МРТ) для постановки диагноза
- обработка спутниковых снимков
- в киноиндустрии

Цветовая коррекция изображений

- Изменение цветового баланса
 - Компенсация:
 - Неверного цветовосприятия камеры
 - Цветного освещения



Яркостные преобразования в киноиндустрии



Основная задача

- улучшение визуального восприятия
- приведение изображения к виду, более удобному для машинного анализа.

Яркостное преобразование

Яркостное преобразование — это операция, которая ставит в соответствие каждому пикселю исходного изображения с интенсивностью **f** новый пиксель с интенсивностью **g** по определенному правилу:

$$g(x, y) = T[f(x, y)],$$

где T – функция преобразования.

Ключевые понятия:

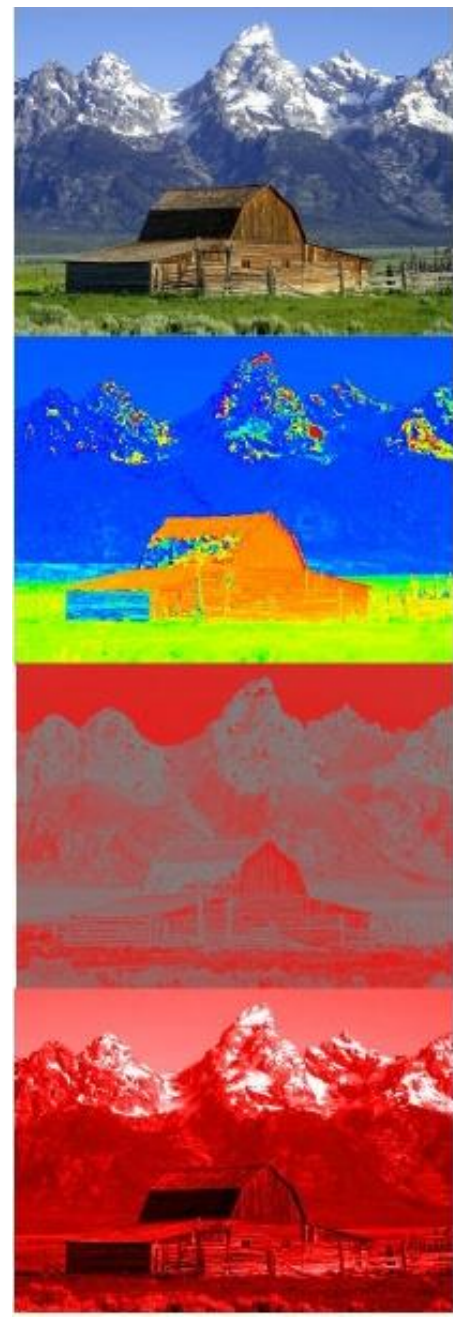
- входное изображение (f),
- выходное изображение (g),
- функция преобразования (T),
- гистограмма изображения.

Изображения

- Многоканальные
- Полутоновые
- Бинарные

Обработка цветных изображений

- По каналам RGB
- По каналам HSL (вначале преобразовать из RGB в HSL)



Подходы к обработке цветных изображений

- Покомпонентная обработка
- Векторная обработка

$$\mathbf{c} = \begin{bmatrix} c_R \\ c_G \\ c_B \end{bmatrix} = \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Преобразование цветного к полутоновому

$$\text{Gray} = (R + G + B) / 3$$

$$\text{Gray} = 0.3 * R + 0.59 * G + 0.11 * B$$

$$\text{Gray} = 0.21 * R + 0.72 * G + 0.078 * B$$

Сравни



NTSC RGB

sRGB

$$Y = 0.3R + 0.59G + 0.11B$$

$$Y = 0.21R + 0.72G + 0.07B$$

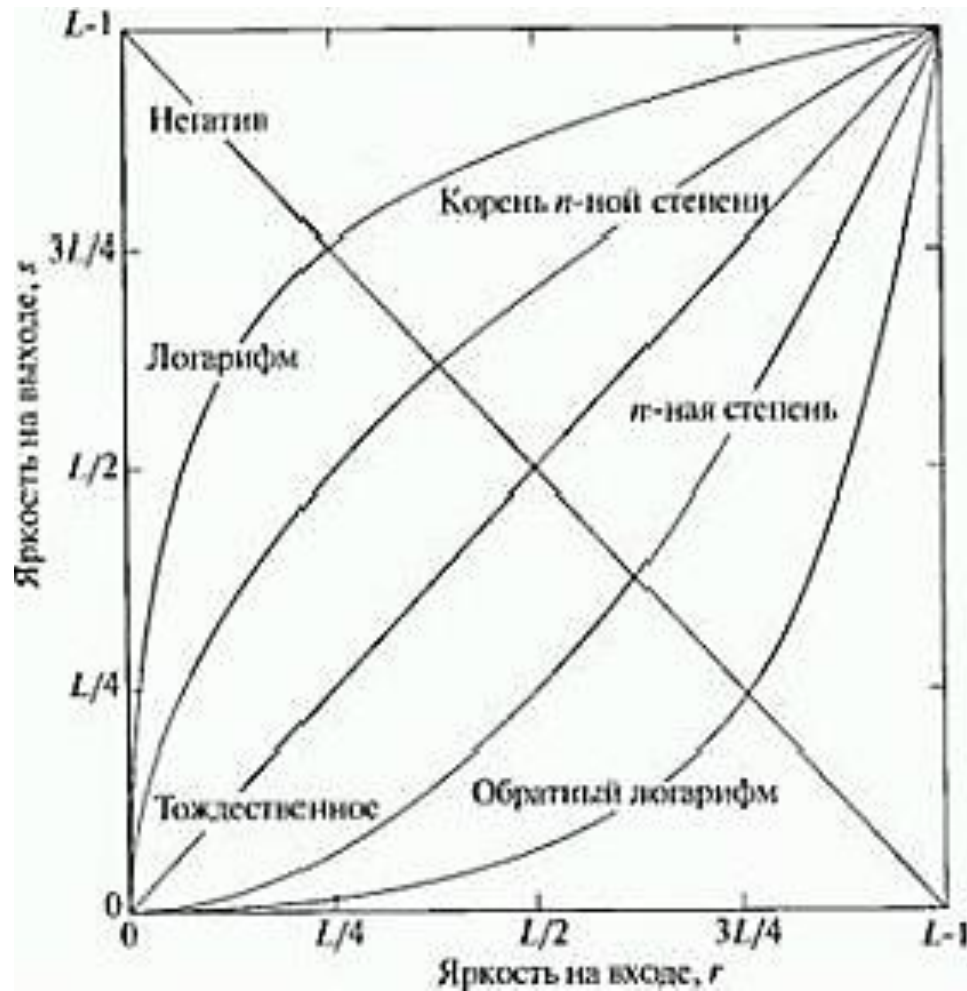
Цветное → полутоновое → бинарное



Классификация яркостных преобразований

- Линейные преобразования (прямая пропорциональность)
- Нелинейные преобразования:
 - степенные
 - логарифмические
 - и т.д.
- Преобразования на основе гистограммы (адаптивные, использующие статистику всего изображения)

Линейные и нелинейные яркостные преобразования



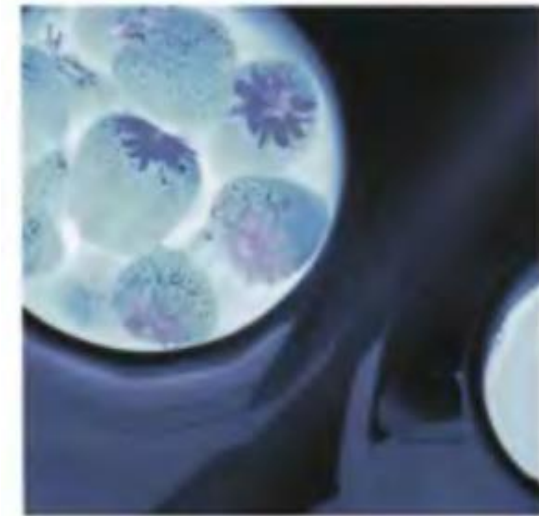
Линейные преобразования

- Тожественное преобразование: $g = f$
(Базовый случай)
- Негатив (инверсия): $g = L - 1 - f$,
где L – максимальное значение яркости (например, 255)
- Изменение яркости и контрастности: $g = a * f + b$
 - Коэффициент a (угол наклона) отвечает за контраст ($a > 1$ – увеличение, $a < 1$ – уменьшение).
 - Коэффициент b (сдвиг) отвечает за яркость ($b > 0$ – увеличение, $b < 0$ – уменьшение).

Цветовое дополнение

$$\text{neg} = \text{Max} - \text{pos}$$

- Аналогично негативу для полутонного изображения
- Полезно для выявления деталей внутри темных областей
- Применение:
 - в медицине (рентген),
 - астрономии
 - и т.д.



Коррекция с опорным цветом

- Предположение
 - Пользователь указывает цвет вручную;
- Источник:
 - Априорные знания – «облака – белые»
 - Хорошая фотография этой же сцены
- Метод
 - Преобразовать по каждому из каналов цвета по формуле:

$$R * \frac{R_{dst}}{R_{src}}; \quad G * \frac{G_{dst}}{G_{src}}; \quad B * \frac{B_{dst}}{B_{src}};$$

Коррекция с опорным цветом

- Примеры:



«Серый мир»

- Предположение:
 - Сумма всех цветов на изображении естественной сцены дает серый цвет;

- Метод:
 - Посчитать средние яркости по всем каналам:

$$\bar{R} = \frac{1}{N} \sum R(x, y); \quad \bar{G} = \frac{1}{N} \sum G(x, y); \quad \bar{B} = \frac{1}{N} \sum B(x, y); \quad Avg = \frac{\bar{R} + \bar{G} + \bar{B}}{3};$$

- Масштабировать яркости пикселей по следующим коэффициентам:

$$R' = R \cdot \frac{Avg}{\bar{R}}; \quad G' = G \cdot \frac{Avg}{\bar{G}}; \quad B' = B \cdot \frac{Avg}{\bar{B}};$$

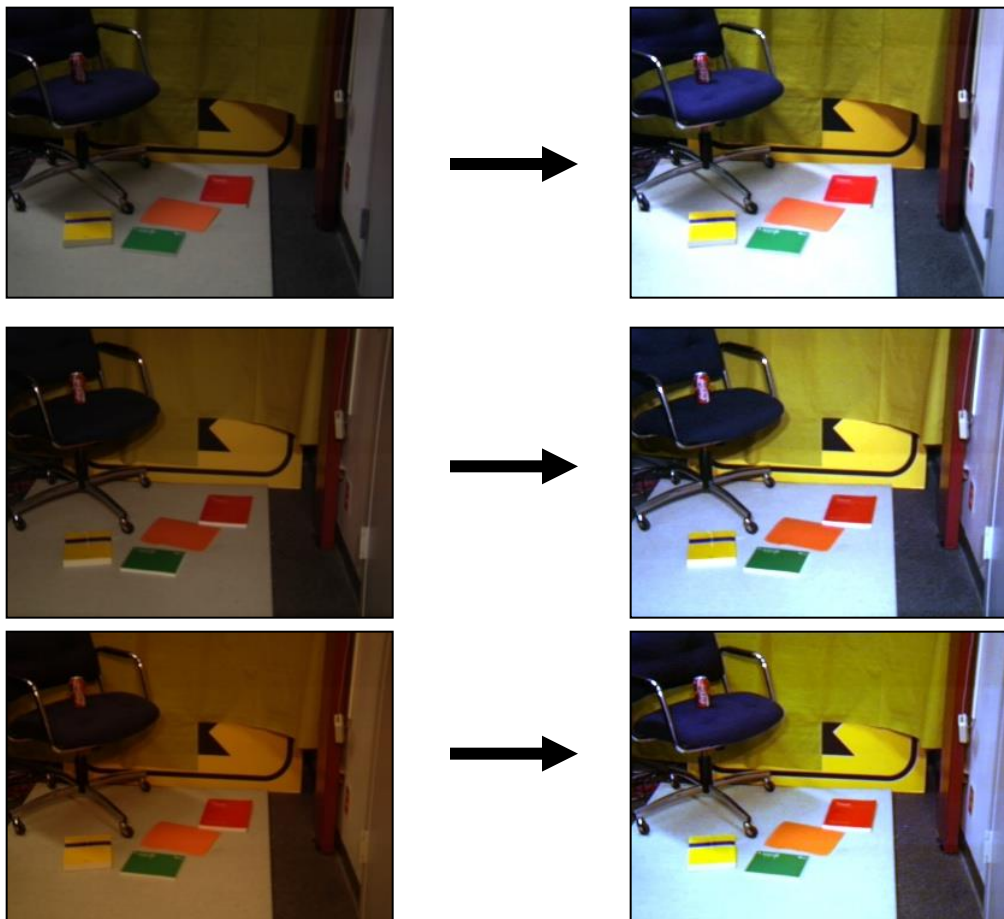
«Серый мир» - примеры



«Серый мир» - примеры

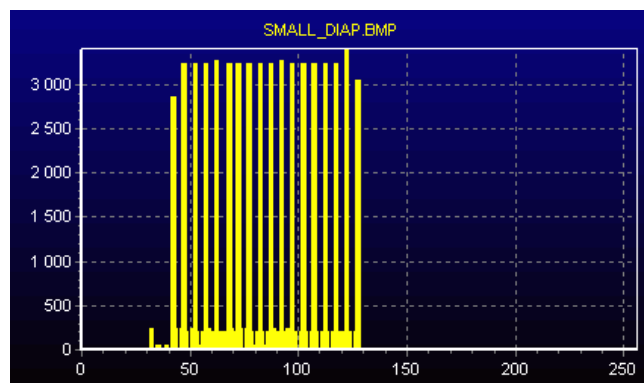


«Серый мир» - примеры



Линейная коррекция яркости

Компенсация узкого диапазона яркостей – линейное растяжение:



$$f^{-1}(y) = (y - y_{\min}) * \frac{(255 - 0)}{(y_{\max} - y_{\min})}$$

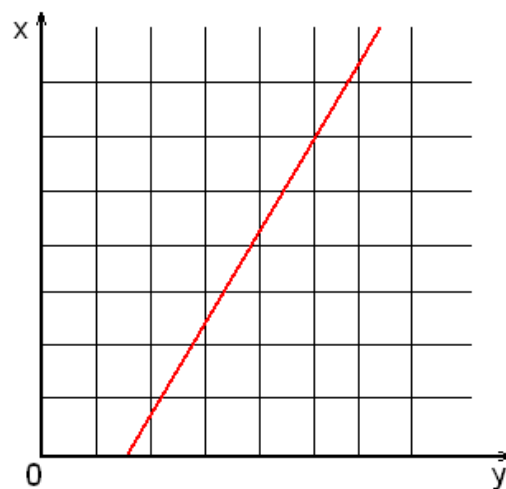
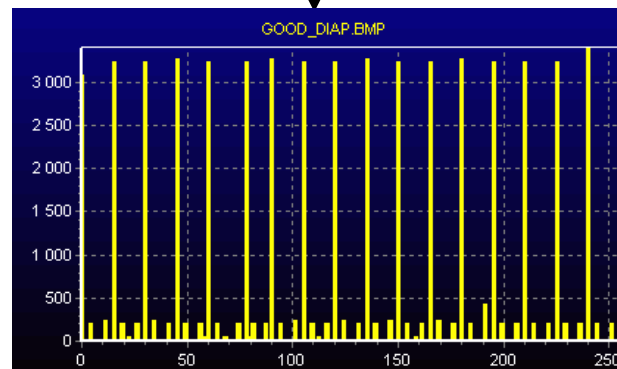
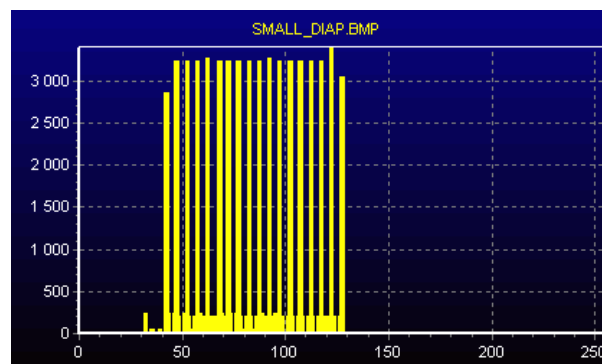
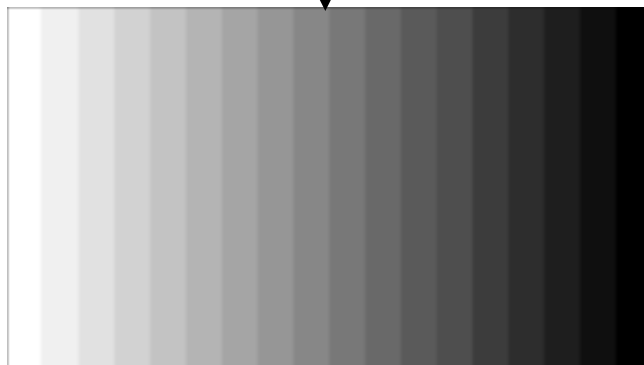


График функции $f^{-1}(y)$

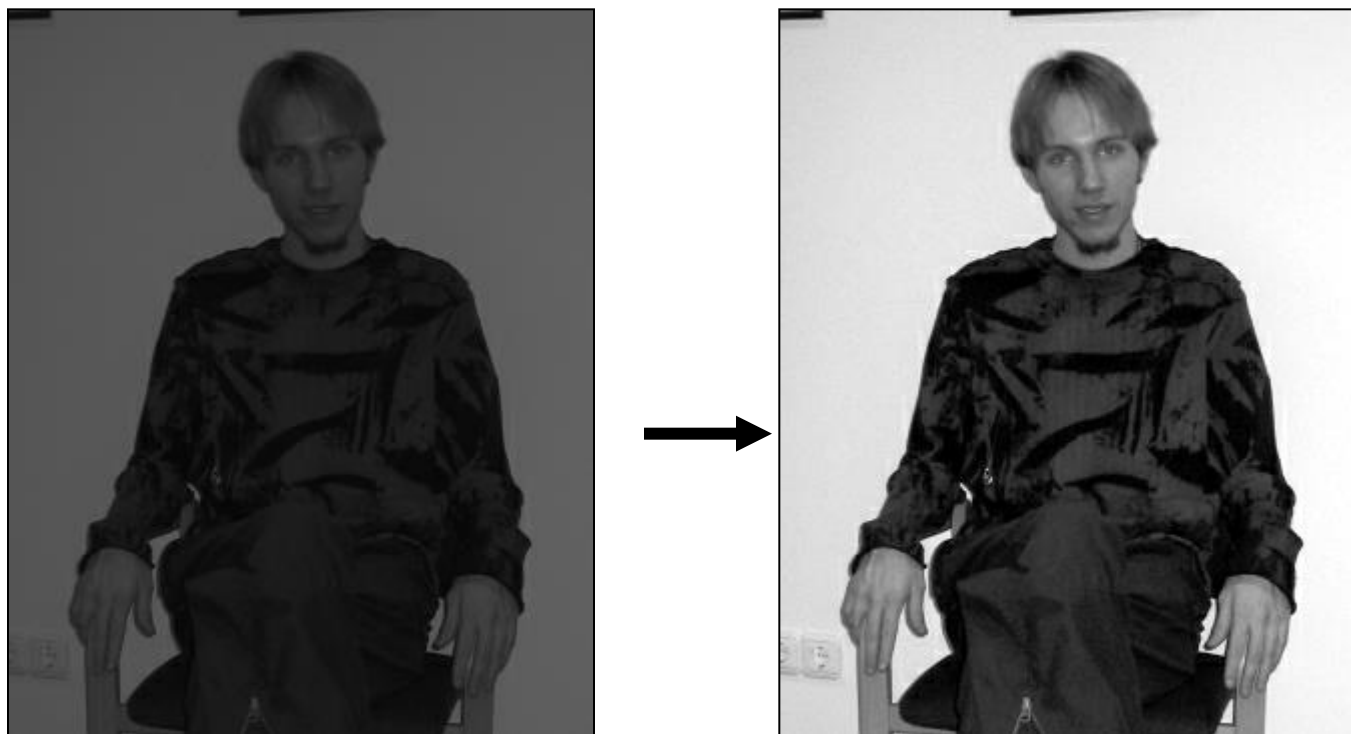
Линейная коррекция яркости

Компенсация узкого диапазона яркостей – линейное растяжение:



Линейная коррекция яркости

Линейное растяжение – «как AutoContrast в Photoshop»



Линейная коррекция



Линейная коррекция помогает не всегда!

Нелинейные преобразования

- Степенные преобразования (Гамма-коррекция)
- Логарифмические преобразования

Степенные преобразования

Степенные преобразования или Гамма-коррекция $y = c \cdot x^\gamma$

Параметр γ (гамма) определяет тип преобразования.

- $\gamma < 1$: «проявление теней» – растягивание темных участков, сжатие светлых
Пример: коррекция слишком темного изображения
- $\gamma > 1$: «проявление света» – растягивание светлых участков, сжатие темных
Пример: повышение резкости деталей в ярких областях
- $\gamma = 1$: линейное преобразование

Демонстрация гамма-коррекции изображения

$$y = c \cdot x^\gamma$$



Логарифмическое преобразование

$$g = c * \log(1 + f)$$

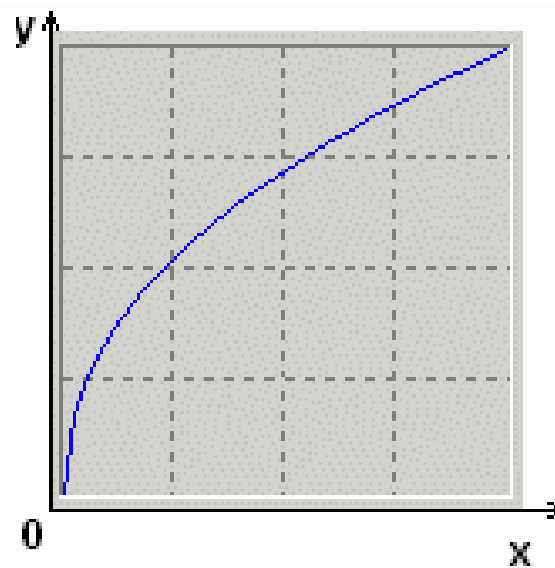
Эффект:

сильное растягивание темных пикселей и сжатие светлых

Применение:

обработка изображений с очень широким динамическим диапазоном (HDR), например, вывод фона и деталей в окне на одном снимке.

Нелинейная коррекция



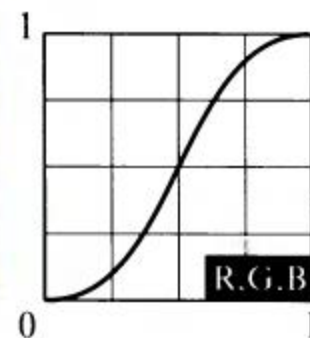
Яркостная коррекция



Малоконтрастное изображение



Результат коррекции



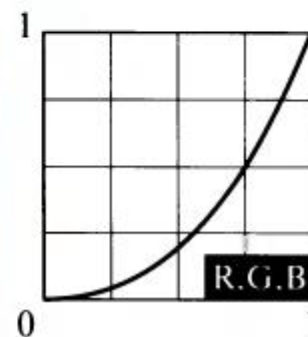
Яркостная коррекция



Светлое изображение



Результат коррекции



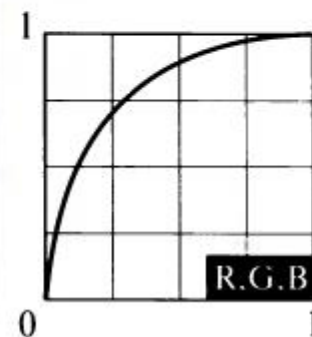
Яркостная коррекция



Темное изображение

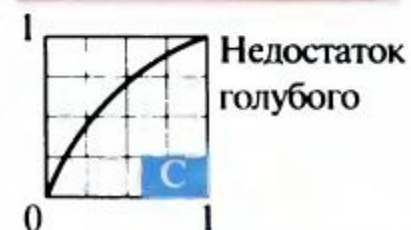
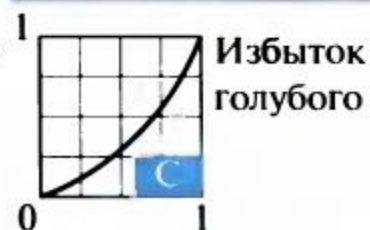
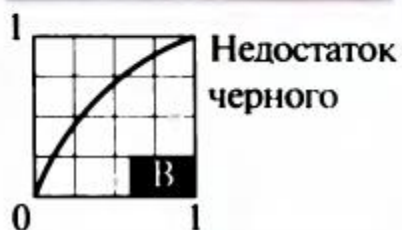
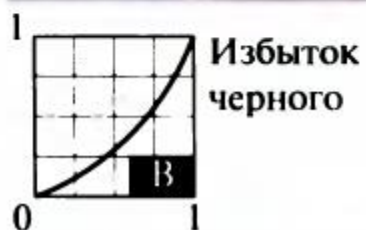


Результат коррекции





Цветовая коррекция



Кусочно-линейные преобразования

- пороговые преобразования
- бинаризация

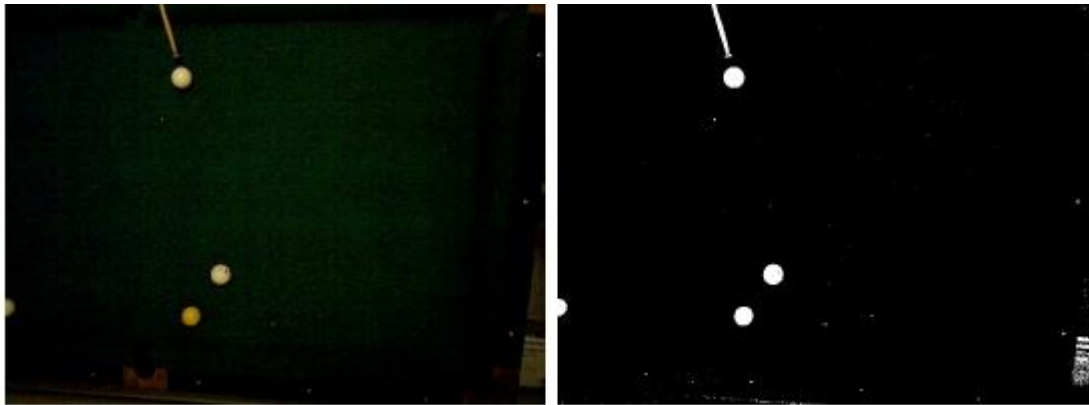
Общая идея кусочно-линейных функций:

Разные правила преобразования для разных диапазонов яркости.

Пороговые преобразования

- бинаризация изображения по одному или двум порогам
- сегментация изображения на несколько яркостных диапазонов
- вырезание определенных диапазонов яркости
- вырезание битовых плоскостей

Пороговое преобразование



$f(v)=1$, если $v>t$

t – порог

$f(v)=0$, иначе

Пороговая бинаризация полутоновых изображений

- Верхняя пороговая бинаризация
- Нижняя пороговая бинаризация
- Пороговая бинаризация по диапазону

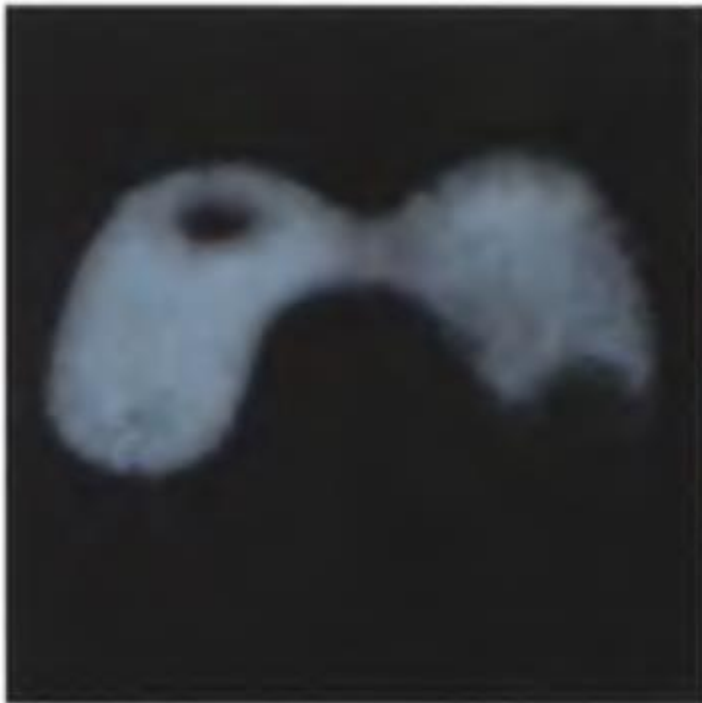
Вырезание цветового диапазона

$$s_i = \begin{cases} 0,5, & \text{если } \left[|r_j - a_j| > \frac{W}{2} \right] \text{ для любого } 1 \leq j \leq n; \\ r_i, & \text{в остальных случаях} \end{cases} \quad i = 1, 2 \dots n$$

$$s_i = \begin{cases} 0,5, & \text{если } \sum_{j=1}^n (r_j - a_j)^2 > R_0^2; \\ r_i, & \text{в остальных случаях;} \end{cases} \quad i = 1, 2 \dots n$$



Квантование по яркости



Применение бинаризации

- распознавание текста (OCR)
- сегментация объектов
- анализ микроскопических изображений
- любая задача, где нужно отделить объект от фона

Главная сложность бинаризации

Главная сложность бинаризации — не применение преобразования, а правильный выбор порога T .

Проблема реализации яркостных преобразований

Многие функции преобразования (особенно нелинейные, как гамма-коррекция) требуют вычисления степеней или логарифмов для каждого пикселя изображения

Для изображения размером 10 мегапикселей это 10 миллионов операций возведения в степень. Это вычислительно дорого

Решение: концепция LUT

LUT (Look-Up Table, или таблица поиска)

LUT — это не отдельный вид преобразования, а механизм (способ) эффективной реализации практически любого яркостного преобразования

LUT — это заранее рассчитанный массив (таблица), где индексом является входное значение яркости f , а значением в ячейке — результат преобразования $g = T(f)$.

Размер LUT:

Для 8-битного изображения (256 градаций) размер LUT будет всего 256 элементов.

Задание LUT

- таблица отображения
- математическая функция
- адаптивное задание по гистограмме

Применение LUT (Look-Up-Table)

$$g(x, y) = T[f(x, y)]$$

$$Im[i, j] = LUT [Im[i, j]]$$

LUT: Osiris и Impulz, представленные компанией «VisionColor»



Преимущества LUT

Высокая скорость:

Операция подстановки из массива по индексу — одна из самых быстрых операций в вычислительной технике.

Это позволяет применять сложные нелинейные преобразования в реальном времени даже на слабых устройствах (камерах, смартфонах).

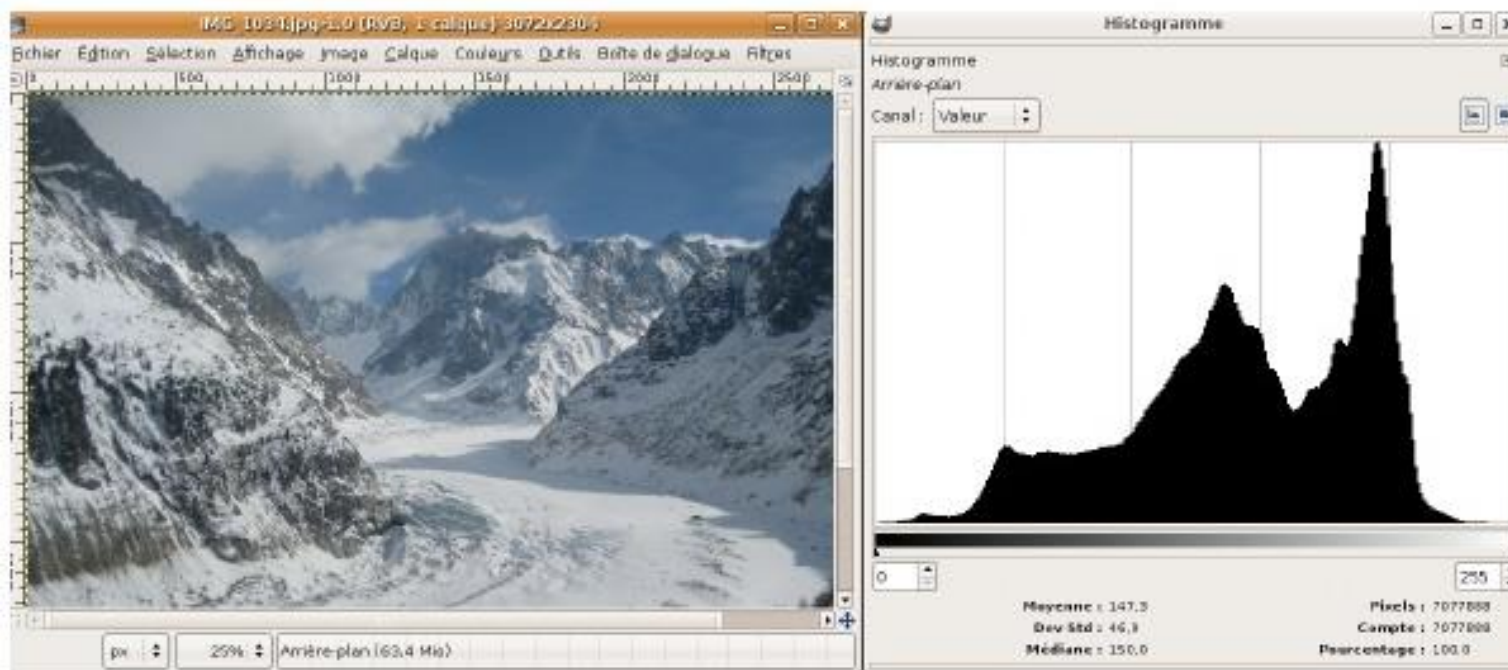
Универсальность:

В LUT можно записать любую функцию преобразования, даже самую сложную и не имеющую аналитического описания (например, кривую, нарисованную пользователем в фоторедакторе).

Стабильность:

Гарантирует, что одно и то же входное значение всегда даст одинаковый выходной результат.

Гистограмма изображения

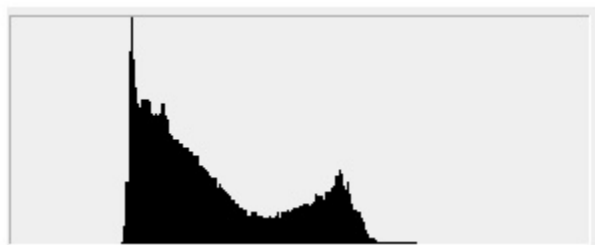


$$h(m) = |\{(r, c) \mid I(r, c) = m\}|$$

Яркостная нормализация изображения



Исходное



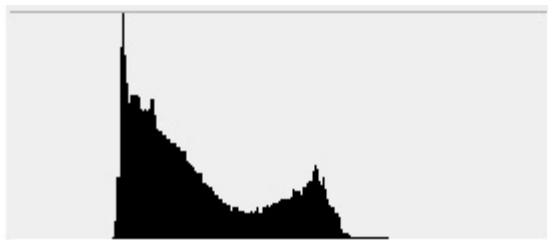
Нормализованное



Эквализация гистограммы



Исходное



С эквализованной гистограммой



Сравнение нормализации и эквализации



Нормализованное



С эквализованной гистограммой



Эквализация гистограммы

- Цель – максимальное соответствие равномерному закону распределения

Эквализация гистограммы

- Значение яркости 0:

$$\text{sum}(\text{hist}[i], i=0..i(0)) \leq q$$

- Значение яркости 1:

$$\begin{aligned} \text{sum}(\text{hist}[i], i=i(0)+1..i(1)) &\leq q && \text{или} \\ \text{sum}(\text{hist}[i], i=0..i(1)) &\leq 2q && \text{и т.д.} \end{aligned}$$

$$q = \frac{w * h}{N}, \quad N = 256$$

- Значение яркости n:

$$\begin{aligned} \text{sum}(\text{hist}[i], i=i(n-1)+1..i(n)) &\leq q && \text{или} \\ \text{sum}(\text{hist}[i], i=0..i(n)) &\leq nq \end{aligned}$$

```
const k = 255;
h: array [0 .. k] of double;

//построение гистограммы
for i := 0 to k do h[i] := 0;

for i := 0 to Height - 1 do
  for j := 0 to Width - 1 do
    h[round(k * Pixels[i, j])] := h[round(k * Pixels[i, j])] + 1;

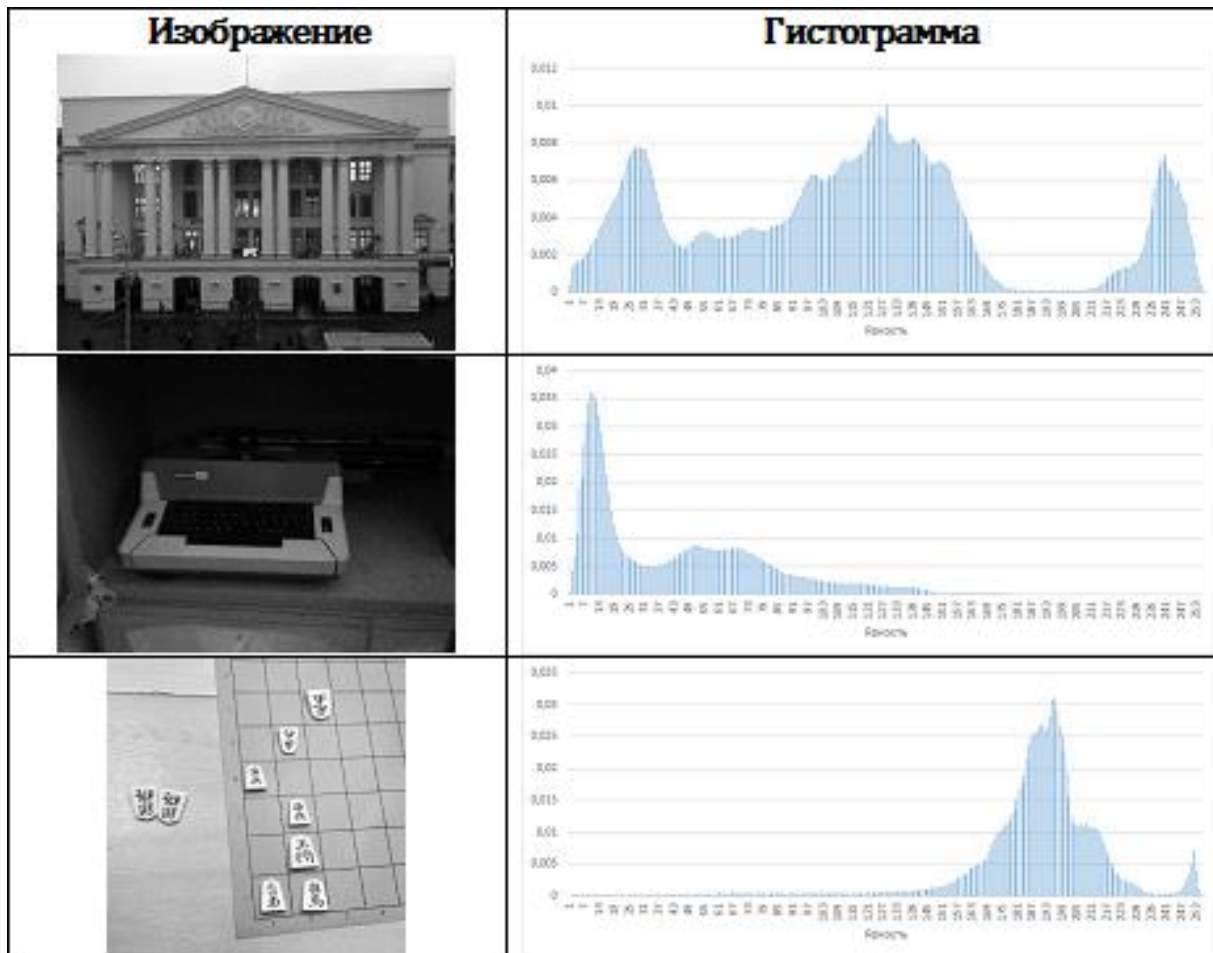
//нормирование гистограммы
for i := 0 to k do
  h[i] := h[i] / (Height * Width);

// построение гистограммы с накоплением
for i := 1 to k do
  h[i] := h[i - 1] + h[i];
```



```
// равномерное распределение значений
for i := 0 to Height - 1 do
  for j := 0 to Width - 1 do
    Pixels[i, j] := h[round(k * Pixels[i, j])];
```

Изображения и их гистограммы



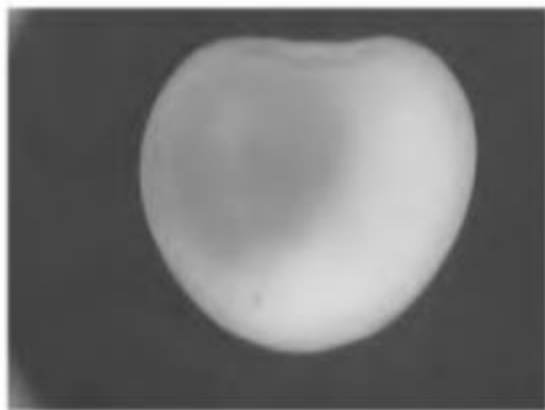
Изображения и их гистограммы после эквализации



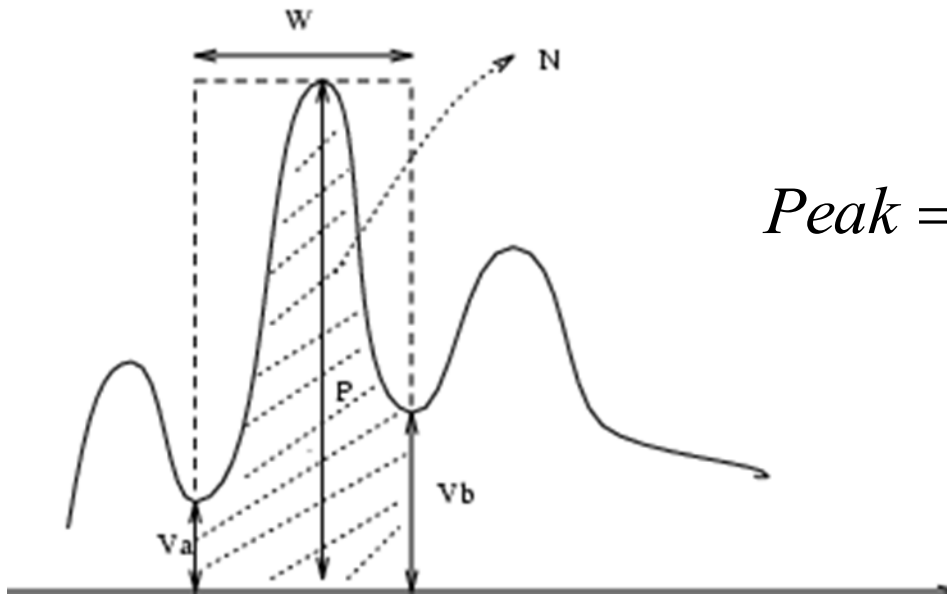
Эквализация для различных цветовых моделей



Бимодальная гистограмма



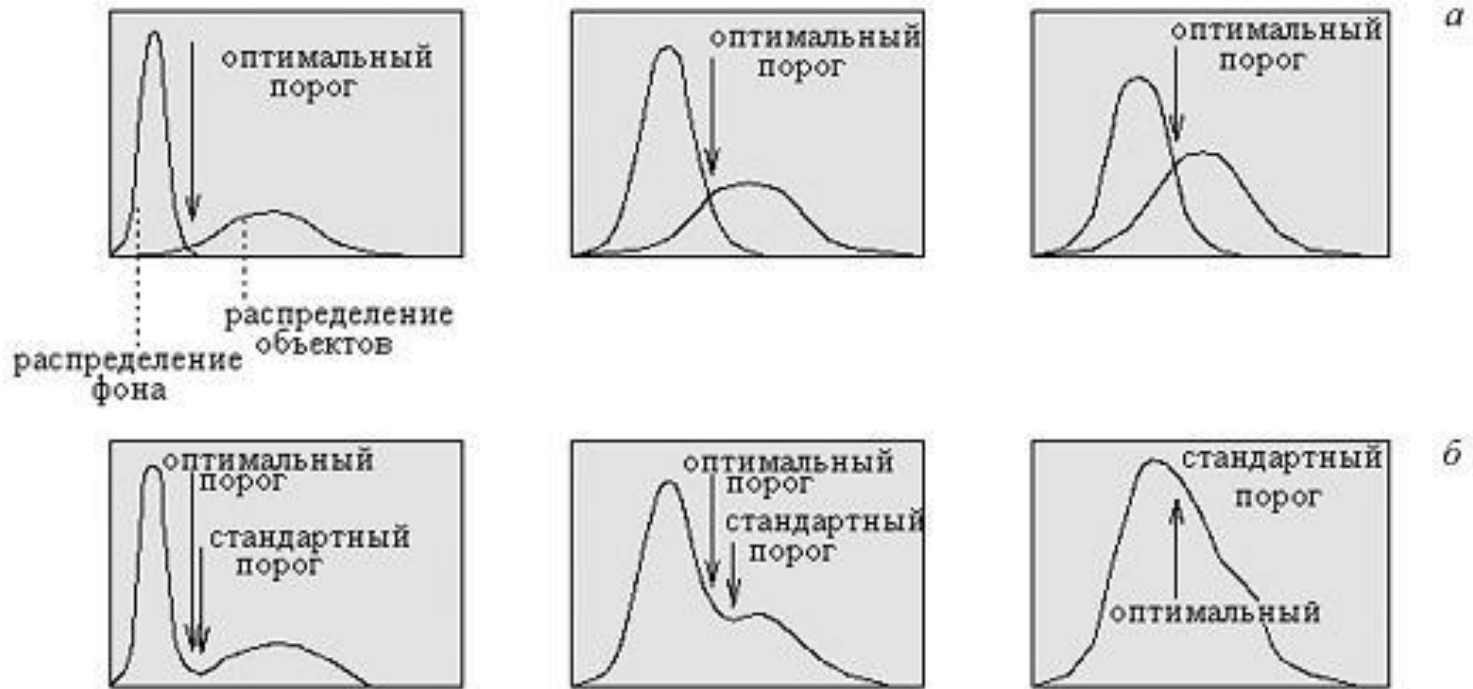
Поиск пиков в гистограмме



$$Peak = \left(1 - \frac{(V_a + V_b)}{2P}\right) \cdot \left(1 - \frac{N}{(W \cdot P)}\right)$$

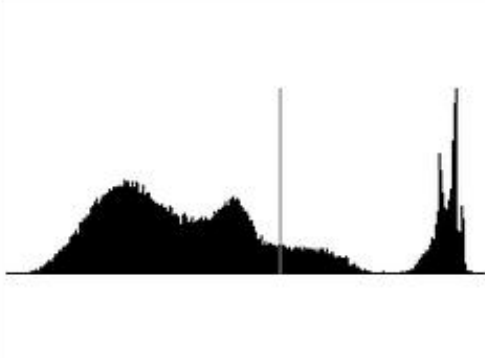
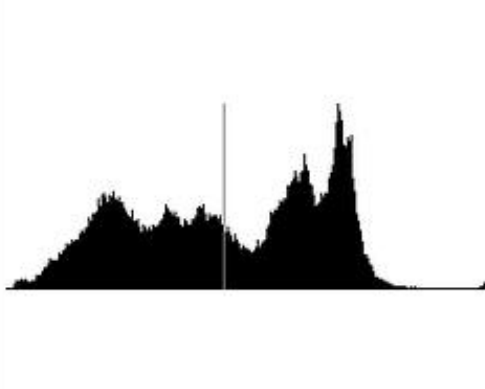
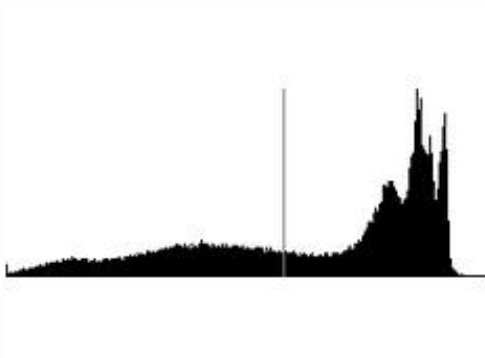
1. Найти соседние локальные максимумы в гистограмме g_i
2. Рассчитать меру «пиковости» для g_i
3. Отфильтровать пики со слишком маленькой «пиковостью».
4. Для оставшихся найти самые «низкие» точки между пиками – это и будут пороги.

Выбор порога бинаризации по гистограмме



а - функции распределения объекта и фона;

б - соответствующие гистограммы и оптимальный порог

Greyscale Image	Binary Image	Histogram
		
		
		

Вручную подобранные пороги



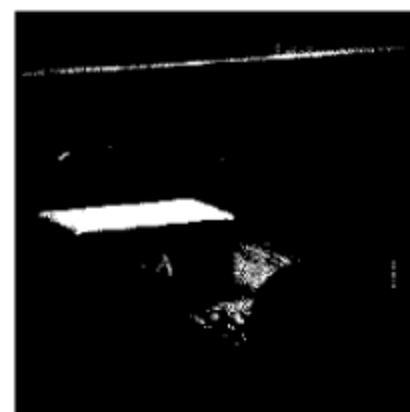
Threshold range 0 to 30



Threshold range 31 to 100



Threshold range 101 to 179



Threshold range 180 to 239

Метод Оцу (Otsu)

- Минимизация внутриклассовой дисперсии
- Максимизация межгрупповой дисперсии

$$sc(t) = 1 - \frac{disp(0, t) + disp(t + 1, \max t)}{disp(0, \max t)}$$

$$T = \arg \max_{t \in 0.. \max t} sc(t)$$



Метод Оцу (Otsu) 1

Минимизация внутригрупповой дисперсии

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

q_1 и q_2 — вероятности первого и второго классов соответственно.

$$\sigma_1^2(t) = \sum_{i=1}^t \frac{[i - \mu_1(t)]^2 P(i)}{q_1(t)} \quad q_1(t) = \sum_{i=1}^t P(i) \quad \mu_1(t) = \sum_{i=1}^t \frac{iP(i)}{q_1(t)}$$

$$\sigma_2^2(t) = \sum_{i=t+1}^T \frac{[i - \mu_2(t)]^2 P(i)}{q_2(t)} \quad q_2(t) = \sum_{i=t+1}^T P(i) \quad \mu_2(t) = \sum_{i=t+1}^T \frac{iP(i)}{q_2(t)}$$

Метод Оцу (Otsu) 2

Максимизация межгрупповой дисперсии

$$\sigma_B^2(t) = q_1(t)[1 - q_1(t)][\mu_1(t) - \mu_2(t)]^2$$

$$q_1(t) = \sum_{i=1}^t P(i)$$

$$q_2(t) = \sum_{i=t+1}^T P(i)$$

$$\mu_1(t) = \sum_{i=1}^t \frac{iP(i)}{q_1(t)}$$

$$\mu_2(t) = \sum_{i=t+1}^T \frac{iP(i)}{q_2(t)}$$

Метод Оцу (Otsu) - Быстрый

Рекуррентные соотношения

$$q_1(t+1) = q_1(t) + P(t+1) \quad \mu = q_1(t)\mu_1(t) + q_2(t)\mu_2(t)$$

$$q_1 = P(1)$$

$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)P(t+1)}{q_1(t+1)} \quad \mu_1(0) = 0$$

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

Быстрый алгоритм метода Оцу

- Вычисляем гистограмму (один проход через массив пикселей). Далее нужна только гистограмма; проходов по всему изображению больше не требуется.
- Начиная с порога $t = 1$, проходим через всю гистограмму, на каждом шаге пересчитывая дисперсию $\sigma_b(t)$. Ищем максимум по $\sigma_b(t)$, запоминая $T = t$.
- Искомый порог равен T .

Пороговое значение определено методом Оцу



a) original image



b) pixels below 93



c) pixels above 93

Метод Оцу (Otsu)

- Достоинства метода Оцу:
 - простота реализации;
 - быстрое время выполнения (требуется $O(N)$ операций, где N — количество пикселей в изображении);
 - нет необходимости подбирать какие-либо коэффициенты, размеры окон для прохода по изображению и проч. (в MatLab это функция `graythresh()` без аргументов);
 - дает предсказуемые результаты.
- Недостатки метода Оцу:
 - не применим для сложных изображений с графикой, цветными надписями, различными градиентами;
 - пороговая бинаризация чувствительна к неравномерной яркости изображения. Решением такой проблемы может быть введение локальных порогов, вместо одного глобального.

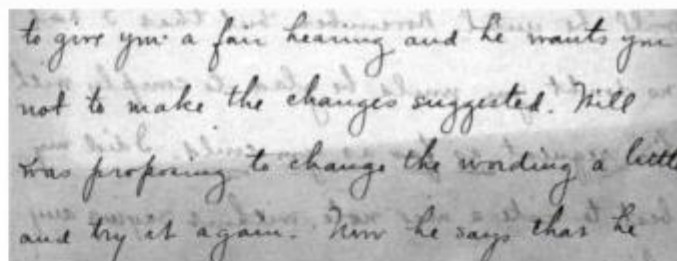
Литература для метода Оцу

- 1. N. Otsu. A threshold selection method from gray-scale histogram. IEEE Transactions on System, Man, and Cybernetics. 9, 62–66, 1979.
- 2. PING-SUNG LIAO, TSE-SHENG CHEN AND PAU-CHOO CHUNG. Fast Algorithm for Multilevel Thresholding. JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 17, 713-727 (2001)

Образцы изображений исторических документов



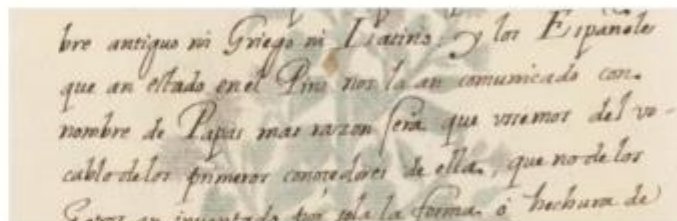
(a) page stain



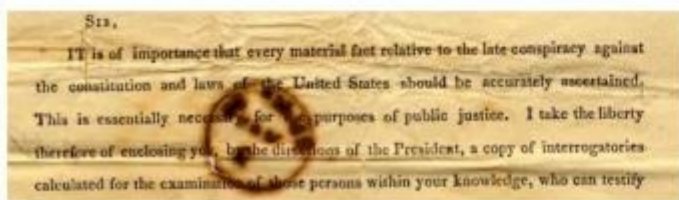
(b) ink bleed through



(c) text stroke fading



(d) background texture



(e) library seal

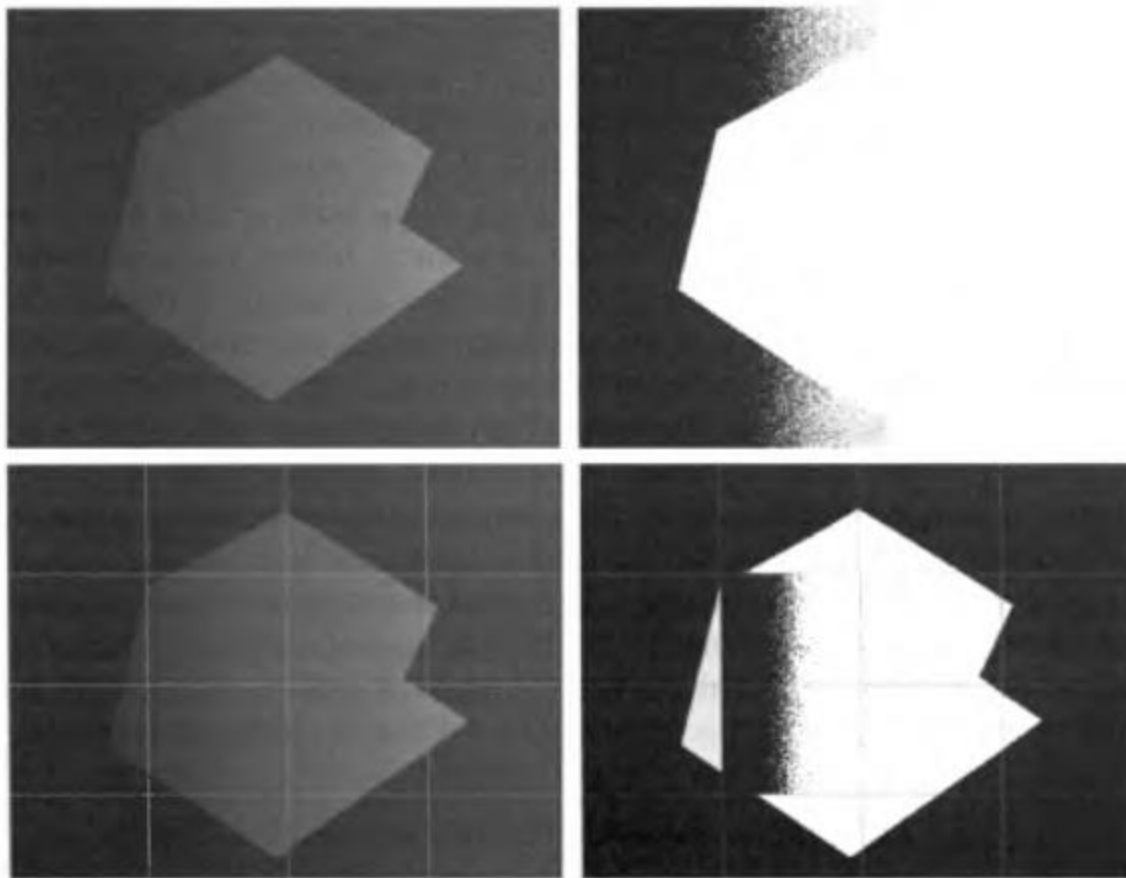


(f) text stroke changes

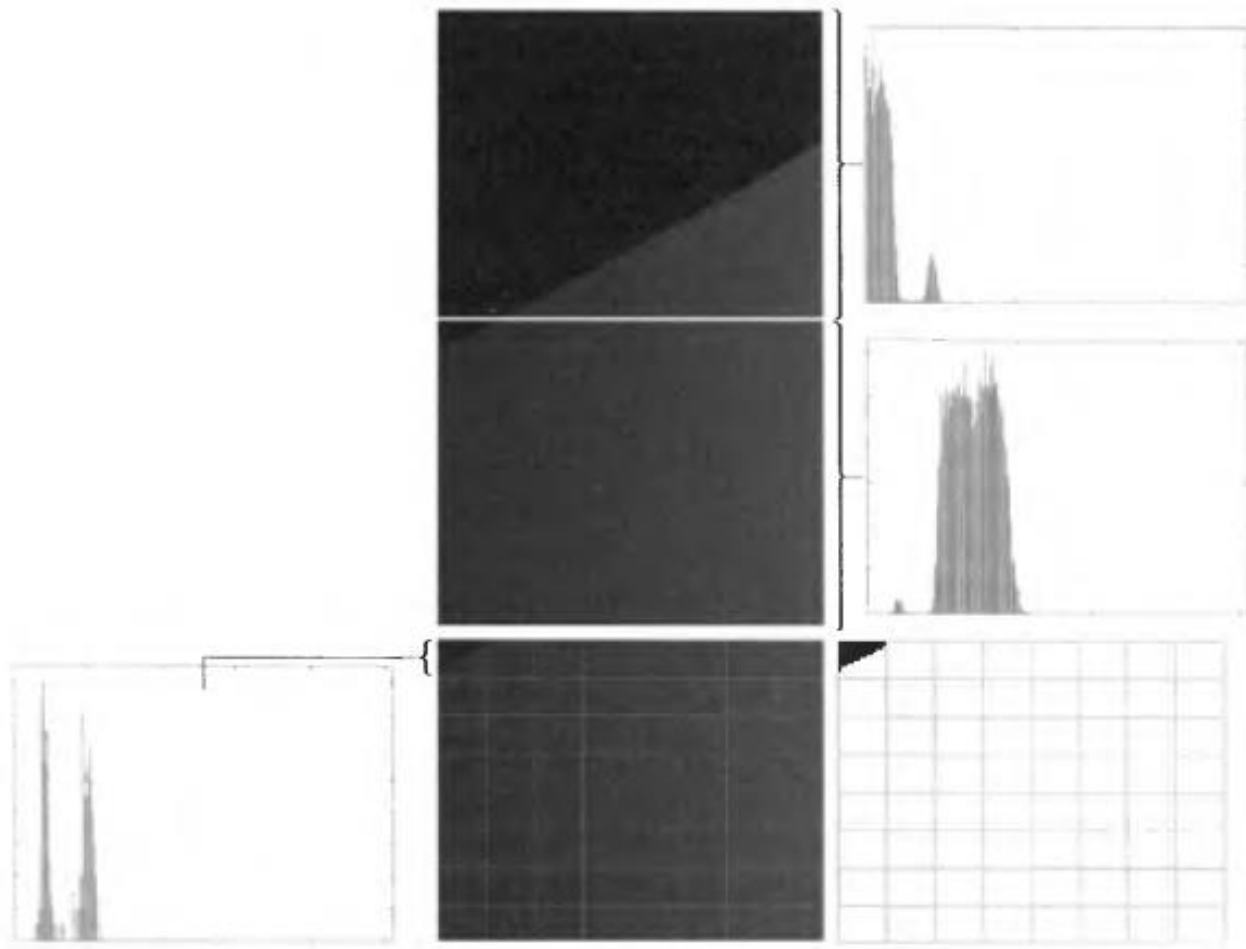
Виды порогов

- Глобальные
- Локальные
- Адаптивные

Глобальные и локальные пороги



Локальные пороги



Какой порог может спасти?



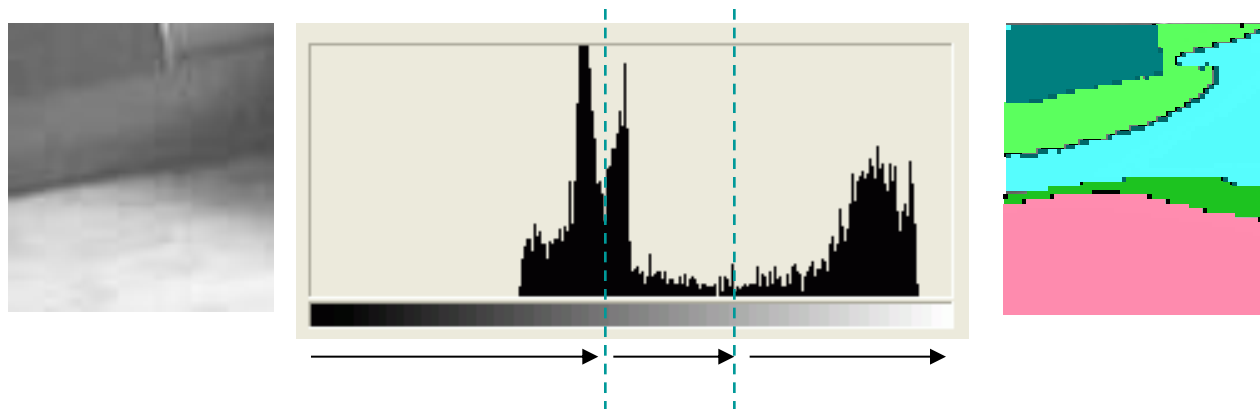
Локальные пороги



Иерархическая бинаризация

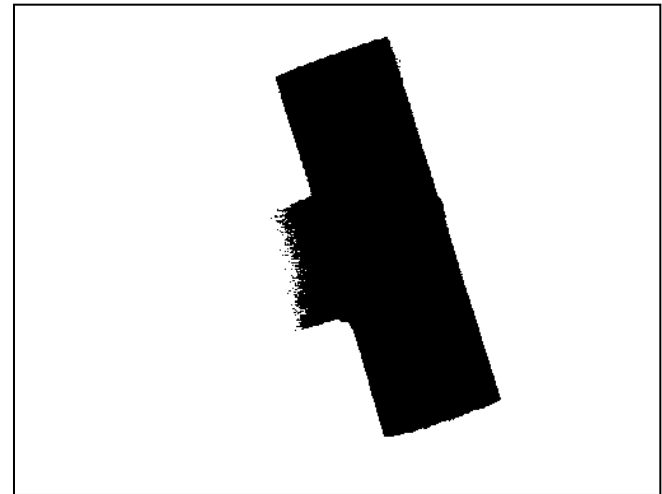
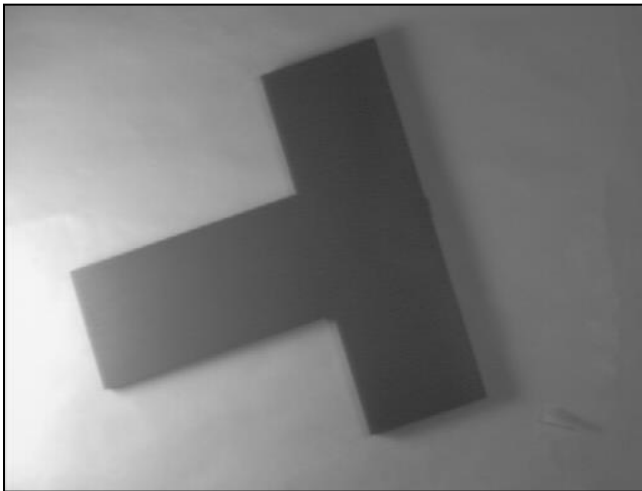


Рекурсивное разделение областей



Адаптивная бинаризация

Необходима в случае неравномерной яркости фона/объекта.



Адаптивная бинаризация

Необходима в случае неравномерной яркости фона/объекта.

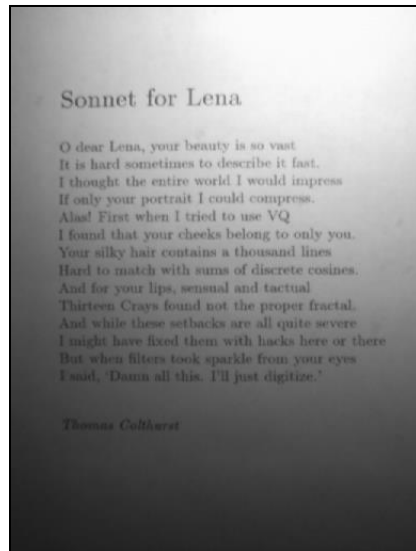
Для каждого пикселя изображения $I(x, y)$:

- В окрестности пикселя радиуса r высчитывается индивидуальный для данного пикселя порог T ;
- Если $I(x, y) > T + C$, результат 1, иначе 0;

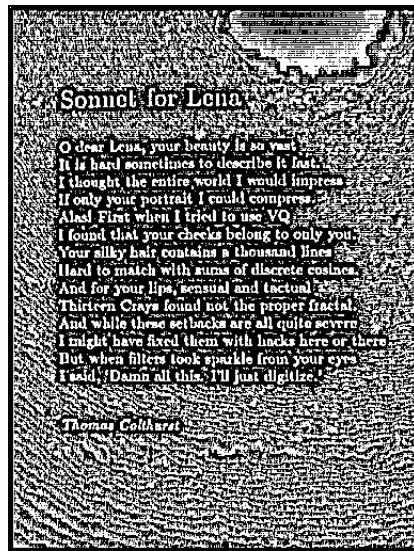
Варианты выбора T :

- $T = mean$
- $T = median$
- $T = (min + max) / 2$

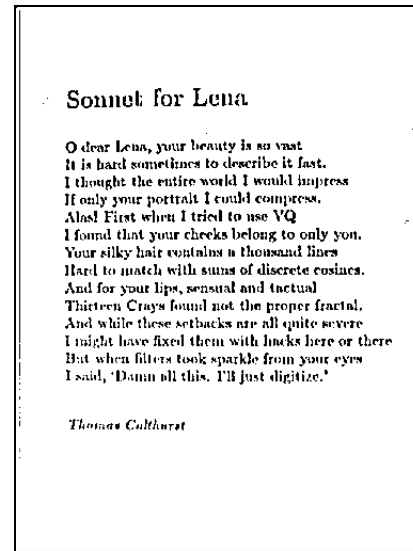
Адаптивная бинаризация



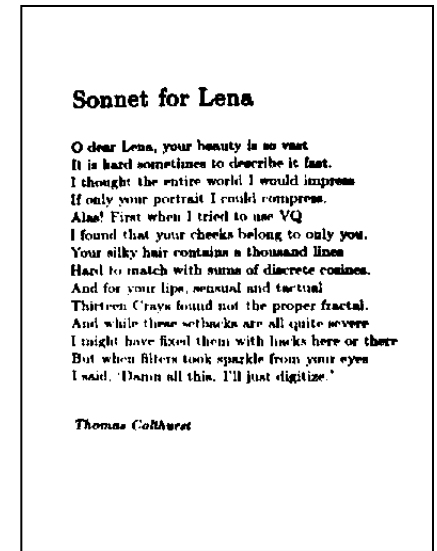
Исходное



$r=7, C=0$



$r=7, C=7$



$r=75, C=10$

Бинаризация :

а), б), в) – метод Оцу;

г), д), е) – алгоритм

Брэдли-Рота;

ж), з), и) –среднее

арифметическое цветов

пикселя;

к), л), м) –

иерархический метод

Оцу

