Государственное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н. Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Р А С Ч Ё Т Н О - П О Я С Н И Т Е Л Ь Н А Я З А П И С К А

к курсовой работе на тему:

Метод сохранения качества изображений при понижении его разрядности

Студент Капустин А.И. И.О. Фамилия
(Подпись, дата)

Руководитель курсового проекта Оленев А.А. И.О. Фамилия
(Подпись, дата)

Москва 2016

Содержание

Глоссарий

Выборка/выборка данных — конечный набор прецедентов (объектов, случаев, событий, испытуемых, образцов, и т.п.), некоторым способом выбранных из множества всех возможных прецедентов, называемого генеральной совокупностью[1].

— Метка(ярлык) - порция данных, идентифицирующая набор данных, описывающая его определенные свойства и обычно хранимая в том же пространстве памяти, что и набор данных[2].

классификатор?

Теория распознавания образа — раздел информатики и смежных дисциплин, развивающий основы и методы классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и т.п. объектов, которые характеризуются конечным набором некоторых свойств и признаков.

Введение

Задача поиска аномалий относится к одному из популярных способов машинного обучения - обучению без учителя. В настоящее время задачу поиска аномалий активно решают во многих областях жизнедеятельности:

а) Защита информации и безопастность

б) Социальная сфера и медицина

в) Банковская и финансовая отрасль

г) Распознавание и обработка текста, изображений, речи

д) Другие сферы деятельности(например, мониторинг неисправностей механизмов)

Задачей поиска выбросов, как частный случай задачи поиска аномалий так же занимаются во вышеперичисленных отраслях.

Количество данных в мире удваивается примерно каждые два года. Поэтому актуальной задачей является разработка новых методов и усовершенствования старых методов поиска выбросов.

В данной работе предлагается новый метод, позволяющий найти аномалии в выборках данных.

1 Аналитический раздел

1.1 Цель и задачи работы

Целью данной работы является создание программного комплекста для обнаружения выбросов временных рядов в собираемых данных. Для достижения данной цели необходимо решить следующие задачи:

— ПЕРЕПИСАТЬ ИЗ ПРЕЗЕНТАЦИИ

— пронализировать предметную область и существующие методы обнаружения выбросов

— разработать метод обнаружения выбросов

— smth

— создать ПО, реализующего разработанный метод обнаружения выбросов

— провести вычислительный эксперименты с использованием разработанного метода

1.2 Обнаружение аномалий

В машинном обучении обнаружение "ненормальных"экземпляров в наборах данных всегда представляло большой интерес. Этот процесс широко известен как обнаружение аномалий или обнаружение выбросов. Вероятно, первое определение было дано Граббсом[3] в 1969 году: "Относительное наблюдение или выброс - это элемент выборки, который, заметно отличается от других членов выборки, в которых он встречается ". Хотя это определение по-прежнему актуально и сегодня, мотивация для обнаружения этих выбросов сейчас совсем другая. Тогда основная причина обнаружения заключалась в том, чтобы удалить выбросы из данных для обучения, поскольку алгоритмы распознавания были весьма чувствительны к выбросам в данных. Эта процедура также называется очищением данных. После разработки более надежных классификаторов интерес к обнаружению аномалий значительно снизился. Однако в 2000 году произошел поворотный момент, когда исследователи стали больше интересоваться самими аномалиями, поскольку они часто связаны с особенно интересными событиями или подозрительными данными. С тех пор было разработано много новых алгоритмов, которые оцениваются в этой статье. В этом контексте определение Граббса также было расширено, так что сегодня аномалии, как известно, имеют две важные характеристики:

а) Аномалия отличается от нормы по своим особенностям

б) Аномалия редко встречается в наборе данных по сравнению с "нормальными"данными

1.2.1 Классификация методов обнаружений аномалий

В отличие от хорошо известной системы классификации, где учебные данные используются для обучения классификатора, а результаты измерений данных оцениваются впоследствии, возможно множество вариантов, когда речь идет об обнаружении аномалий. Метод обнаружения аномалий, которая будет использоваться, зависит от ярлыков, доступных в наборе данных, и мы можем выделить три основных типа:

а) Обучение с учителем. Доступны полностью размеченные данные для обучения и для тестов. Обычный классификатор может быть обучен один раз и применяться впоследствии. Это похоже на традиционное распознавание образов, за исключением того, что классы обычно сильно не сбалансированы. Поэтому не все алгоритмы классификации идеально подходят для этой задачи. Для многих применений аномалии не известны заранее или могут возникать спонтанно в качестве новинок на этапе тестирования.

б) Обучение с частичным привлечением учителя.Обучение использует учебные и тестовые наборы данных. Данные обучения состоят только из нормальных данных без каких-либо аномалий. Основная идея заключается в том, что модель нормального класса изучается, а аномалии могут быть обнаружены впоследствии, отклоняясь от этой модели. Эта идея также известна как «одноклассовая» классификация [4].

в) Обучение без учителя.Самая гибкий способ, который не требует каких-либо меток. Кроме того, нет различия между учебным и тестовым наборами данных. Идея заключается в том, что алгоритм обнаружения аномалии оценивает данные исключительно на основе внутренних свойств набора данных. Как правило, расстояния или плотности используются для оценки того, что является нормальным, а что является выбросом. В этой статье основное внимание уделяется этой неконтролируемой установке обнаружения аномалий.

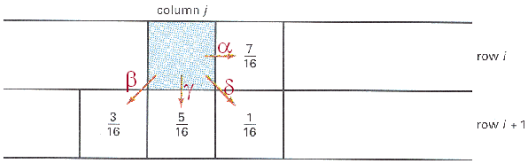
В анализе данных есть два основных направления, которорые занимаются поиском аномалий - это детектирование новизны и обнаружение выбросов. "Новый объект это так же объект, который отличается по своим свойствам от объектов выборки. Однако, в отличие от выброса, он его ещё нет в самой выборке и задача анализа сводится к его обнаружению при появление. Например, если вы анализируете замеры уровня шума и отбрасываетете слишком высокие или слишком низкие значения, то вы боретесь с выбросам. А если Вы создаёте алгоритм, который для каждого нового замера оценивает, насколько он похож на прошлые, и выбрасывает аномальные — вы "боретесь с новизной"[5]. Выбросы являются следствием:

а) ошибок в данных

б) неверно классифицированных объектов

в) присутсвием объектов других выборок

г) намеренным искажением данных



$$\alpha + \beta + \gamma + \delta = 1.0$$

Рисунок 1.1 — Пример размеченного набора данных

На рисунке 1.1 можно увидеть желтые точки - выброс "слабом смысле". Они незначительно отклоняются от основных данных(зеленые точки). Красные же точки являются аномальными - выбросами "в сильном смысле они значительно отклоняются с от основывных данных. В данной работе будет изучаться вопрос находждения "сильных выбросов"и критериев отличия сильного выброса от основных данных. В дальнейшем под словом "выброс"будет подразумеваться "сильный выброс а под аномалией - в выброс(выброс является частным случаем аномалии). Понятие аномалии зачастую интерпетируют по-разному в зависимости от характера данных. Обычно аномалией назыют некоторое отклонение от нормы. Это определение нуждается в формальном уточнении.

1.3 Постановка задачи

Кстати, про картинки. Во-первых, для фигур следует использовать `[ht]`. Если и после этого картинки вставляются «не по ГОСТ», т.е. слишком далеко от места ссылки, — значит у вас в РПЗ **слишком мало текста**! Хотя и ужасный параметр `!ht` у окружения `figure` тоже никто не отменял, только при его использовании документ получается страшный, как в ворде, поэтому просьба так не делать по возможности.

1.4 Существующие подходы к созданию всячины

Известны следующие подходы...

а) Перечисление с номерами.

б) Номера первого уровня. Да, ГОСТ требует именно так — сначала буквы, на втором уровне — цифры. Чуть ниже будет вариант «нормальной» нумерации и советы по её изменению. Да, мне так нравится: на первом уровне выравнивание элементов как у обычных абзацев. Проверим теперь вложенные списки.

1) Номера второго уровня.

2) Номера второго уровня. Проверяем на длиииином-предлиииииииииииинной строке, что получается.... Сойдёт.

в) По мнению Лукьяненко, человеческий мозг старается подвести любую проблему к выбору из трех вариантов.

г) Четвёртый (и последний) элемент списка.

Теперь мы покажем, как изменить нумерацию на «нормальную», если вам этого захочется. Пара команд в начале документа поможет нам.

1) Изменим нумерацию на более привычную...
2) ... нарушим этим гост.
   а) Но, пожалуй, так лучше.

В заключение покажем произвольные маркеры в списках. Для них нужен пакет **enumerate**.

1. Маркер с арабской цифрой и с точкой.
2. Маркер с арабской цифрой и с точкой.
   I. Римская цифра с точкой.
   II. Римская цифра с точкой.

В отчётах могут быть и таблицы — см. табл. 1.1 и 1.2. Небольшая таблица делается при помощи **tabular** внутри **table** (последний полностью аналогичен **figure**, но добавляет другую подпись).

Таблица 1.1 — Пример короткой таблицы с коротким названием

Тело	F	V	E	$F+V-E-2$
Тетраэдр	4	4	6	0
Куб	6	8	12	0
Октаэдр	8	6	12	0
Додекаэдр	20	12	30	0
Икосаэдр	12	20	30	0
Эйлер	666	9000	42	$+\infty$

Для больших таблиц следует использовать пакет **longtable**, позволяющий создавать таблицы на несколько страниц по ГОСТ.

Для того, чтобы длинный текст разбивался на много строк в пределах одной ячейки, надо в качестве ее формата задавать `p` и указывать явно ширину: в мм/дюймах (`110mm`), относительно ширины страницы (`0.22\textwidth`) и т.п.

Можно также использовать уменьшенный шрифт — но, пожалуйста, тогда уж во **всей** таблице сразу.

Таблица 1.2 — Пример длинной таблицы с длинным названием на много длинных-длинных строк

Вид шума	Громкость, дБ	Комментарий
Порог слышимости	0	
Шепот в тихой библиотеке	30	
Обычный разговор	60-70	
Звонок телефона	80	Конечно, это было до эпохи мобильников
Уличный шум	85	(внутри машины)
Гудок поезда	90	
Шум электрички	95	
Порог здоровой нормы	90-95	Длительное пребывание на более громком шуме может привести к ухудшению слуха
Мотоцикл	100	
Power Mower	107	(модель бензокосилки)
Бензопила	110	(Doom в целом вреден для здоровья)
Рок-концерт	115	
Порог боли	125	feel the pain
Клепальный молоток	125	(автор сам не знает, что это)
Порог опасности	140	Даже кратковременное пребывание на шуме большего уровня может привести к необратимым последствиям
Реактивный двигатель	140	
	180	Необратимое полное повреждение слуховых органов
Самый громкий возможный звук	194	Интересно, почему?..

2 Конструкторский раздел

2.1 Оценка качества изображений

2.1.1 PSNR

Пиковое отношение сигнала к шуму (англ. peak signal-to-noise ratio) - соотношение между максимумом возможного значения сигнала и мощностью шума, искажающего значения сигнала.[?]

$$PSNR = 20\log_{10}\frac{MAX_i}{\sqrt{MSE}} \tag{2.1}$$

Где MAX_i - это максимальное значение, принимаемое пикселем изображения, MSE - среднеквадратичное отклонение. Для двух монохромных изображений I и K размера m×n, одно из которых считается зашумленным приближением другого, вычисляется так:

$$MSE = \frac{1}{m*n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|I(i,j) - K(i,j)|^2 \tag{2.2}$$

2.1.2 SSIM

Индекс структурного сходства (SSIM от англ. structure similarity) — метод измерения схожести между двумя изображениями путем полного сопоставления. SSIM-индекс является развитием традиционных методов, таких как PSNR (peak signal-to-noise ratio) и метод среднеквадратичной ошибки MSE, которые оказались несовместимы с физиологией человеческого восприятия.

Отличительной особенностью метода, в отличие от MSE и PSNR, является то, что он учитывает «восприятие ошибки» благодаря учёту структурного изменения информации. Идея заключается в том, что пиксели имеют сильную взаимосвязь, особенно когда они близки пространственно. Данные зависимости несут важную информацию о структуре объектов и о сцене в целом. Особенностью является, что SSIM всегда лежит в промежутке от -1 до 1, причем при его значении равном 1, означает, что мы имеем две одинаковые картинки. Общая формула имеет вид

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(\sigma_x y + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2.3}$$

Тут μ_x среднее значение для первой картинки, μ_y для второй, σ_x среднеквадратичное отклонение для первой картинки, и соотвественно σ_y для второй, $\sigma_x y$ это уже ковариация. Она находится следующим образом:

$$\sigma_x y = \mu_x y - \mu_x\mu_y \tag{2.4}$$

c_1 и c_2 - поправочные коэффициенты, которые нужны вследствие малости знаменателя.

$$c_1 = (0{,}01 * d)^2 \tag{2.5}$$

$$c_2 = (0{,}03 * d)^2 \tag{2.6}$$

d - количество цветов, соответствующих данной битности изображения Для подтверждения или опровержения вышеописанных гипотезы реализуются несколько алгоритмов случайного распределения, алгоритм Флойда-Стейнберга, модификации на основе алгоритма Флойда-Стейнберга. Результаты работы алгоритмов сравниваются по времени работы, по количеству затрачиваемой памяти,а так же по SSIM и PSNR.

2.1.3 Алгоритм случайного распределения

P(x,y) - цвет конкретного пикселя

Листинг 2.1 — Алгоритм случайного распределения

```
for x in range(height):
    for y in range(weight):
        if P(x,y)>127:
            P(x,y) = 255
        else:
            P(x,y) = 0
```

2.2 Виды случайных распределений

2.2.1 Белый шум

Белый шумом называют сигнал с равномерной спектральной плотностью на всех частотах и дисперсией, равной бесконечности. Является стационарным случайным процессом. В качестве сигнала в задаче дизеринга рассматривается последовать последовательность чисел, получаемых от генератора случайных чисел.

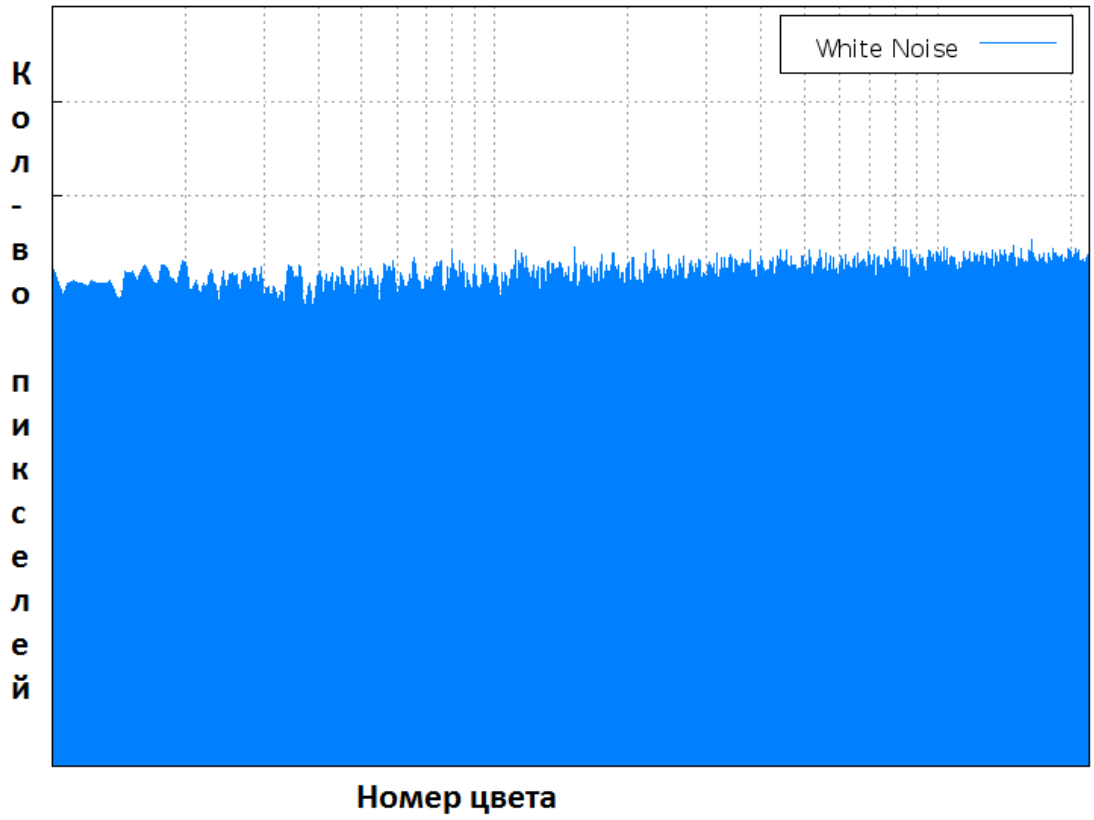


Рисунок 2.1 — Диаграмма белого шума

2.2.2 Коричневый шум

Спектральная плотность коричневого шума пропорциональна $1/f^2$, где f — частота. Это означает, что на низких частотах шум имеет больше энергии, чем на высоких. То есть пикселей темных цветов большей, чем пикселей светлого цвета.Применение фильтра коричневого шума в целом затемняет получаемое изображение.

Листинг 2.2 — Получение коричневого шума

```
1 def smoother(noise):
2     output = []
3     for i in range(len(noise) - 1):
4         output.append(0.5 * (noise[i] + noise[i+1]))
5 return output
```

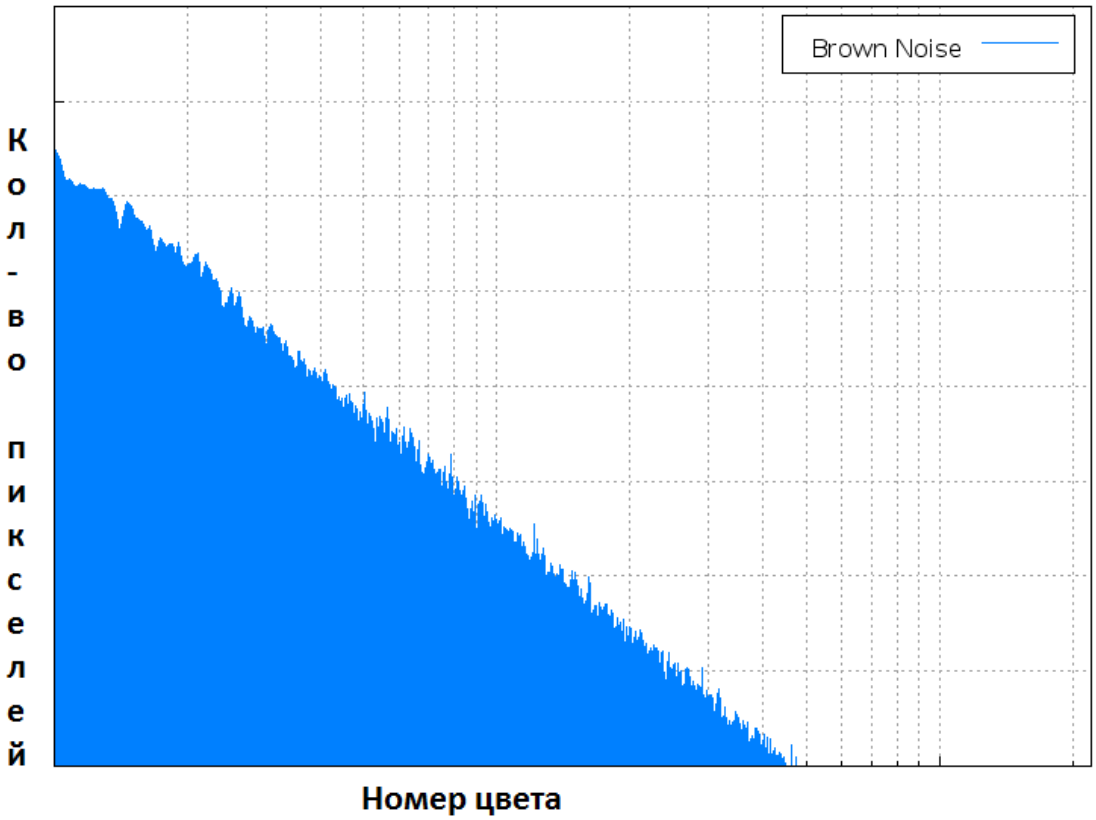


Рисунок 2.2 — Диаграмма красного шума

2.2.3 Гауссовский шум

Гауссовский шум - шум, имеющий функцию плотности вероятности (PDF), равную нормальному распределению, которое также известно как гауссово распределение.

$$p_g(z) = \frac{1}{\sigma\sqrt{2 * \pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \tag{2.7}$$

z - количество цветов, μ -среднее значение, σ - стандартное отклонение.

2.2.4 Фиолетовый шум

Фиолетовый шум представляет собой противооложноть между коричневому шуму. Получение его аналогично получению коричневого шума.Применение фильтра фиолетового шума в целом засветляет получаемое изображение.

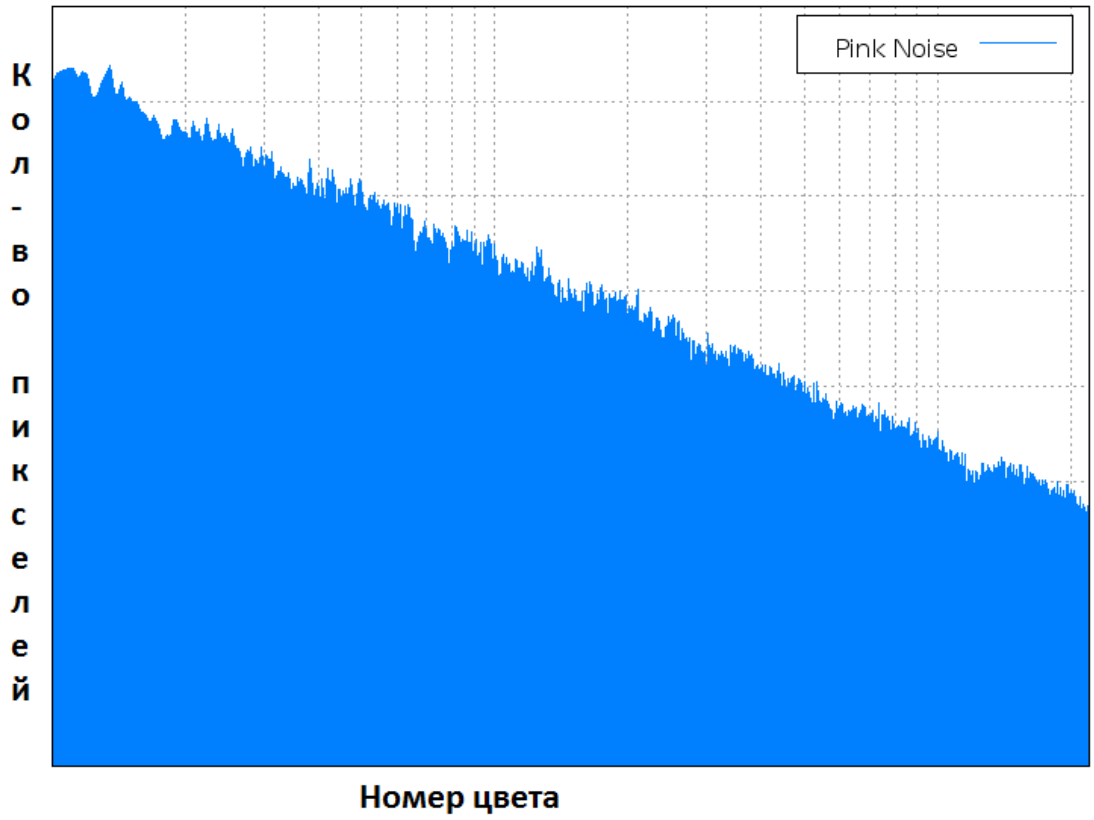


Рисунок 2.3 — Диаграмма розового шума

Листинг 2.3 — Получение розового шума

```
1 def rougher(noise):
2     output = []
3     for i in range(len(noise) - 1):
4     output.append(0.5 * (noise[i] - noise[i+1]))
5 return output
```

2.2.5 Розовый и синий шумы

Розовый и синий шумы представляют собой "промежуточные"шумы.Изобржаение с розовым шумом темнее изображения с белым шумом, но светлее изображения с коричневым шумом. Изображение с синим шумом светлее изображения с белым шумом, но темнее чем изображение с фиолетовым шумом. Их получение аналогично получению со коричневого и фиолетового шумов.

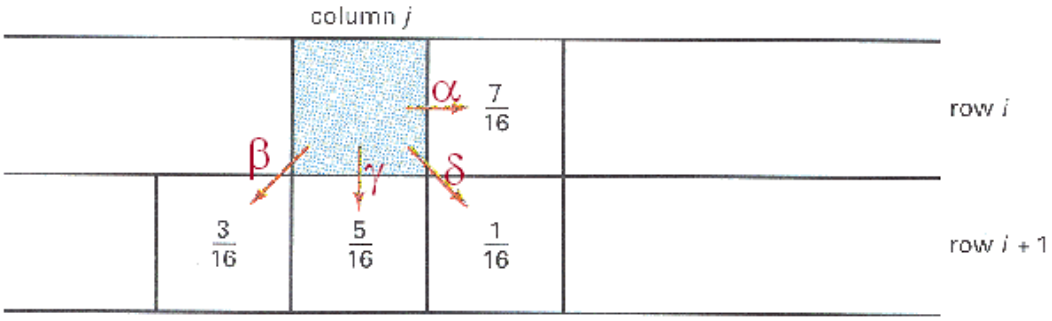
2.3 Алгоритм Байера

Листинг 2.4 — Алгоритм Флойда-Стейнберга

```
For each pixel, Input,:
    Factor  = ThresholdMatrix[xcoordinate % X][ycoordinate % Y];
    Attempt = Input + Factor * Threshold
    Color = FindClosestColorFrom(Palette, Attempt)
```

2.4 Алгоритм Флойда-Стейнберга

Рассмотрим более детально алгоритм Флойда-Стейнберга. P(x,y) - цвет пикселя в точке x,y I(x,y) - предполгаемый цвет пикселя с учетом ошибки(действительное число) Следует отметить, что сумма "долей"рассевания ошибки



$$\alpha + \beta + \gamma + \delta = 1.0$$

Рисунок 2.4 — Схема рассеивания ошибки

и для других алгоритмов упорядоченного дизеринга будет равна единице.

Листинг 2.5 — Алгоритм Флойда-Стейнберга

```
for x in range(width):
    for y in range(height):
        P(x,y) = trunc(I(x,y)+0.5)
        e = I(x,y) - P(x,y)
        I(x,y-1) += 7/16*e
        I(x+1, y-1) += 3/16*e
        I(x+1, y) += 5/16*e
        I(x+1, y+1) += 1/16*e
```

2.5 Ложный алгоритм Флойда-Стейнберга

Отличительной особенностью ложного алгоритма Флойда-Стейнберга является по пикселям изображения справа-налево, а не слева-направо как в большинстве других алгоритмов.

Листинг 2.6 — Ложный алгоритм Флойда-Стейнберга

```
1  for x in range(width):
2      for y in range(height):
3          P(x,y) = trunc(I(x,y)+0.5)
4          e = I(x,y) - P(x,y)
5          P(x,y) = trunc(I(x,y)+0.5)
6          e = I(x,y) - P(x,y)
7          I(x,y-1) += 3/8*e
8          I(x-1, y-1) += 2/8*e
9          I(x-1, y) += 3/8*e
```

2.6 Алгоритм Джарвиса,Джунка и Нинка

Листинг 2.7 — Алгоритм Джарвиса,Джунка и Нинка

```
1  for x in range(width):
2      for y in range(height):
3          P(x,y) = trunc(I(x,y)+0.5)
4          e = I(x,y) - P(x,y)
5          P(x,y) = trunc(I(x,y)+0.5)
6          e = I(x,y) - P(x,y)
7          I(x,y-1) += 3/48*e
8          I(x+1,y+1) += 5/48*e
9          I(x,y+1) += 7/48*e
10         I(x-1,y+1) += 5/48*e
11         I(x-2,y+1) += 3/48*e
12         I(x-1,y+1) += 5/48*e
13         I(x+2,y+2) += 1/48*e
14         I(x+1,y+2) += 3/48*e
15         I(x, y+2) += 5/48*e
16         I(x-1,y+2) += 3/48*e
17         I(x-2,y+2) += 1/48*e
18         I(x+1,y) += 7/48*e
19         I(x+2,y) += 5/48*e
```

2.7 Первый алгоритм Юлиомы

Листинг 2.8 — Первый алгоритм Юлиомы

```
For each pixel, Input, in the original picture:
    Factor  = ThresholdMatrix[xcoordinate % X][ycoordinate % Y];
    Make a Plan, based on Input and the Palette.

    If Factor < Plan.ratio,
        Draw pixel using Plan.color2
    else,
        Draw pixel using Plan.color1
```

Листинг 2.9 — План нахождения цвета пикселя

```
SmallestPenalty = 10^99
For each unique combination of two colors from the palette, Color1
   and Color2:
    For each possible Ratio, 0 .. (X*Y-1):
        Mixed = Color1 + Ratio * (Color2 - Color1) / (X*Y)
        Penalty = Evaluate the difference of Input and Mixed.

        If Penalty < SmallestPenalty,
        SmallestPenalty = Penalty
        Plan = { Color1, Color2, Ratio / (X*Y) }1
```

Листинг 2.10 — Вспомогательная матрица перого Алгоритма Юлиомы

```
    [0/64, 48/64, 12/64, 60/64, 3/64, 51/64, 15/64, 63/64,
    32/64, 16/64, 44/64, 28/64, 35/64, 19/64, 47/64, 31/64,
    8/64, 56/64, 4/64, 52/64, 11/64, 59/64, 7/64, 55/64,
    40/64, 24/64, 36/64, 20/64, 43/64, 27/64, 39/64, 23/64,
    2/64, 50/64, 14/64, 62/64, 1/64, 49/64, 13/64, 61/64,
    34/64, 18/64, 46/64, 30/64, 33/64, 17/64, 54/64, 29/64,
    10/64, 58/64, 6/64, 54/64, 9/64, 57/64, 5/64, 53/64,
    42/64, 26/64, 38/64, 22/64, 41/64, 25/64, 37/64, 21/64]
```

3 Технологический раздел

3.1 Выбор языка программирования

Для реализации данных алгоритмов был выбран язык C++. Данный язык был обоснован следующими причинами: Причины:

1) Компилируемый язык со статической типизацией.
2) Сочетание высокоуровневых и низкоуровневых средств.
3) Реализация ООП.
4) Наличие удобной стандартной библиотеки шаблонов

3.2 Выбор вспомогательных библиотек

Для реализации программы была выбрана библиотека Qt.

1) Широкие возможности работы с изображениями, в том числе и попиксельно
2) Наличии более функциональных аналогов стандартной библиотеки шаблонов в том числе для разнообразных структур данных

Так же были были использованы библиотеки ImageMagick(для конвертации изображения в ограниченную цветовую палитру), OpenMP(многопоточность), OpenGL(работа с шейдерами)

3.2.1 Диаграмма классов

Для реализации различных алгоритмов была разработана следующая структура классов. Был создан абстракный класс дизеринга Dithering с интерфейсом, наследуюмом в дочерних классах. Так же была введена система менджеров: DitherManager, MetricsManager, DataManager и MainManager.

4 Исследовательский раздел

4.1 Время дизеринга раличных алгоритмов

Рассмотрим время работы различных алгоритмов для различных размеров изображения.

	Размер, пиксели	Время, мкс
White noise	133x90	862
Blue noise	133x90	930
Brown noise	133x90	934
Violet noise	133x90	937
Pink noise noise	133x90	930
Floyd-SD	133x90	1200
F. Floyd-SDe	133x90	1093
JJN	133x90	1909
White noise	458x458	15735
Blue noise	458x458	19374
Brown noise	458x458	19432
Violet noise	458x458	18787
Pink noise noise	458x458	18129
Floyd-SD	458x458	27173
F. Floyd-SDe	458x458	26424
JJN	458x458	47201
White noise	458x458	194376
Blue noise	458x458	200577
Brown noise	458x458	208400
Violet noise	458x458	251294
Pink noise noise	458x458	258775
Floyd-SD	458x458	251294
F. Floyd-SDe	458x458	387104
JJN	458x458	857481

Из рассмотрения вынесены алгоритм Юлиомы в вследствие того, что он значительно медленней других алгоритмов(2732568 мкс для изображения 113x90) в и алгоритм Байера, реализованный при помощи шейдеров, вследствии того, что он не не укладывается в рамки требуемой палитры (при этом он работает очень быстро 64 мс для изображени 640x480).

4.2 Качество получаемого изображения

	PSNR	SSIM
White noise	33.2894	0.914778
Blue noise	36.1756	0.971626
Brown noise	33.32370	0.915767
Violet noise	37.63480	0.984574
Pink noise	36.4484	0.974718
Floyd-SD	37.0553	0.979173
F. Floyd-SDe	36.8401	0.976452
JJN	37.30740	0.981688
Yliouma	36.2359	0.967796
Without dithering	37.6348	0.984574

Несмотря на то, что некоторые сложные алгоритмы дизеринга диффузии ошибок обещат получения хорошего качества изображений,некоторые алгоритмы случайного дизеринга на конкретных изображениях дают лучший результат. Для того чтобы получить наилучший результат дизеринга, следует проанализировать результаты дизеринга нескольких изображений и выбрать среди них наилучшее. Так же следует отметить некоторую необъективность метрик: результат метрик не всегда совпадает с человеческим восприятием картинки.

4.3 Размер получаемого изображения

	Разрешение, пикс	Размер, кб	Исх. раз., кб
White noise	900x675	186	2373 bmp,1779 png
Blue noise	900x675	135	
Brown noise	900x6750	186	
Violet noise	900x675	98	
Pink noise	900x675	1158	
Floyd-SD	900x675	1273	
F. Floyd-SDe	900x675	143	
JJN	900x675	117	
White noise	3984x32355	3431	50344 bmp,37758 png
Blue noise	3984x3235	2570	
Brown noise	3984x3235	3432	
Violet noise	3984x3235	1950	
Pink noise	3984x3235	2406	
Floyd-SD	3984x32355	3605	
F. Floyd-SDe	3984x3235	4269	
JJN	3984x3235	3716	

Из вышеприведенной таблицы, можно заметить, размер изображения после дизеринга значительно уменьшиается,достирается выигрыш в размере изображение до 15 раз, в зависимости от иходного контейнера изображения и выбранного способра дизеринга.

Заключение

В данной работе были реализованы различные алгоритмы дизеринга, было произведено сравнение и анализ этих алгоритмов. Программа позволяет получить изображение схожего визуального качества при значительном уменьшении размера. Был получен вывод о том, что для различных целей следует использовать различные алгоритмы дизеринга, универсального алгоритма дизеринга не существует. Программа не привязана к какой-то конкретной операционной системе и может быть скомпилирована и запущена на всех популярных ОС.

Список использованных источников

1. *machinelearning.ru/*. Выборка. https://goo.gl/7gjJ6p.

2. *определения, ГОСТ 20886-85: Организация данных в системах обработки данных. Термины и.* 11. http://www.gostrf.com/normadata/1/4294832/4294832686.pdf.

3. *F.E., Grubbs.* Procedures for Detecting Outlying Observations in Samples. Technometrics / Grubbs F.E. — 1969.

4. *Moya M.M., Hush D.R.* Network Constraints and Multi-objective Optimization for One-class Classification. Neural Networks / Hush D.R. Moya M.M. — 1996.

5. *Дьяконов, Александр.* Поиск аномалий (Anomaly Detection) / Александр Дьяконов. — 2017. https://goo.gl/Z43Ne9.