

# Содержание

<b>1</b>	<b>Обзор методов шумоподавления</b>	<b>6</b>
1.1	Постановка задачи . . . . .	6
1.2	Модели шума . . . . .	6
1.2.1	Гауссовский шум . . . . .	7
1.2.2	Пуассоновский шум . . . . .	8
1.3	Простейшие алгоритмы . . . . .	9
1.3.1	Вох фильтр . . . . .	10
1.3.2	Гауссовский фильтр . . . . .	11
1.3.2.1	Достоинства и недостатки "box" фильтра и фильтра гаусса . . . . .	12
1.3.3	Медианный фильтр . . . . .	12
1.3.3.1	Достоинства и недостатки медианного фильтра . . . . .	13
1.4	Обзор современных методов шумоподавления . . . . .	14
1.5	Билатеральный фильтр . . . . .	14
1.5.1	Основная идея . . . . .	14
1.5.2	Описание . . . . .	14
1.5.3	Результат работы фильтра . . . . .	15
1.6	Guided фильтр . . . . .	16
1.6.1	Основная идея . . . . .	16
1.6.2	Описание . . . . .	16
1.6.3	Результат работы фильтра . . . . .	18
1.7	Non-Local mean фильтр . . . . .	18
1.7.1	Основная идея . . . . .	18
1.7.2	Описание . . . . .	18
1.7.3	Результат работы фильтра . . . . .	19
1.8	BM3D . . . . .	20
1.8.1	Оценка дисперсии . . . . .	20

1.8.2	Первый шаг . . . . .	21
1.8.2.1	Группировка . . . . .	21
1.8.2.2	Совместная фильтрация . . . . .	22
1.8.2.3	Агрегация . . . . .	22
1.8.3	Второй шаг . . . . .	23
1.8.3.1	Группировка . . . . .	23
1.8.3.2	Совместная фильтрация . . . . .	23
1.8.3.3	Агрегация . . . . .	24
1.8.4	Оптимальные параметры . . . . .	24
1.8.5	Результат работы фильтра . . . . .	24
1.9	Total Variation фильтр . . . . .	25
1.9.1	Основная идея . . . . .	25
1.9.2	Описание . . . . .	25
1.9.3	Результат работы фильтра . . . . .	26
1.10	Заключение . . . . .	27
<b>2</b>	<b>Критерии оценки эффективности алгоритмов</b>	<b>28</b>
2.1	PSNR . . . . .	28
2.2	SSIM . . . . .	28
2.2.1	Выбор оптимального алгоритма . . . . .	29
2.3	Заключение . . . . .	30
<b>3</b>	<b>Исследования</b>	<b>31</b>
3.1	Тестовое множество . . . . .	31
3.1.1	Изображения . . . . .	31
3.1.2	Видео . . . . .	32
3.2	Схема программы . . . . .	32
3.3	Результаты исследования . . . . .	33

д35

# ВВЕДЕНИЕ

В результате формирования, передачи и преобразовании с помощью электронных систем, изображения могут быть подвержены различным искажениям, что в ряде случаев ухудшает качество изображения, с визуальной точки зрения, а также скрывает некоторые участки изображений. Актуальность задач шумоподавления изображений, полученных с помощью фотокамер, видеокамер, рентгеновских снимков или другим не искусственным методом, растет с каждым годом по мере развития информационных технологий.

На текущем этапе современных средств компьютерной техники можно выделить несколько направлений. Распознавание образов - обнаружение на изображении объектов с определенными характеристиками, свойственных некоторому классу объектов. Обработка изображений - преобразует некоторым способом изображение. Визуализация - генерирование изображения на основе некоторого описания. Важную роль играют системы автоматизации всех этих процессов.

Первостепенной задачей такой системы является улучшение качества изображения. Это в первую очередь достигается за счет уменьшения количества шума на изображениях. На данный момент не существуют универсальных алгоритмов, которые позволят это сделать. Поэтому исследования в этой области не прекращаются и по сей день.

Чаще всего шумоподавление служит для улучшения визуального восприятия, но может также использоваться для каких-то специализированных целей — например, в медицине для увеличения четкости изображения на рентгеновских снимках, в качестве предварительной обработки для последующих алгоритмов. Также шумоподавление играет важную роль при сжатии изображений. При сжатии сильный шум может быть принят за детали изображения, и это может отрицательно повлиять на результирующее качество.[1]

Целью данной квалификационной работы является изучение следующих алгоритмов шумоподавления: BM3D, Non-Local Means, Guided, Bilateral, Total Variation.

Так же необходимо провести сравнительный анализ данных алгоритмов.

Дипломная работа состоит из двух разделов. В первом разделе произведен обзор алгоритмов шумоподавления, а так же некоторых моделей шума. Во второй части приведена информация о тестовом множестве, классификации изображений и видео, описание методов оценки эффективности методов шумоподавления.

# 1 Обзор методов шумоподавления

## 1.1 Постановка задачи

Будем считать, что существует исходное изображение  $x$ , которое не искажено шумом. После процедуры искажения шумом  $n$ , получаем шумное изображение  $y$ . Степень зашумленности изображения можно описать следующим понятием SNR. SNR (отношение сигнал шум) - безразмерная величина, равная отношению мощности сигнала на мощность шума. В текущих обозначениях получим:

$$SNR = \frac{P_y}{P_n} \quad (1)$$

где:

- $P_x$  - мощность сигнала (изображения)
- $P_n$  - мощность шума

Значит, что бы уменьшить влияния шума на изображения, необходимо либо увеличить мощность сигнала, что из-за особенностей строения устройств, детектирующих изображение, приведет к увеличению шума. Либо уменьшить количество шума. Вторым методом и занимаются алгоритмы шумоподавления. В данной ВКР будем считать, что изображение  $x$  будет искажено аддитивным шумом  $n$ , в результате чего мы получим изображение  $y$ . Опишем это следующим образом.

$$y = x + n \quad (2)$$

Целью шумоподавления является нахождения исходного изображения  $x$ , при этом слагаемое  $n$  неизвестно, а доступно лишь только зашумленное изображение  $y$ .

## 1.2 Модели шума

Шум изображения - это случайное изменение яркости или цветовой информации на изображениях и, как правило, аспект электронного шума. Шум может

как зависеть от изображения так и быть независимым. Является нежелательным побочным продуктом захвата изображения, который скрывает желаемую информацию.

Шум изображения может варьироваться от практически незаметных пятен на цифровой фотографии, сделанной при хорошем освещении, до оптических и радиоастрономических изображений, которые почти полностью представляют собой шум, из которого небольшое количество информации может быть получено с помощью сложной обработки. Такой уровень шума был бы недопустим на фотографии, так как было бы невозможно даже определить объект.[2]

Характер проблемы удаления шума зависит от типа шума, повреждающего изображение. Рассмотрим некоторые типы шума:

- Гауссовский шум
- Пуассоновский шум
- Импульсный шум
- Экспоненциальный шум

### 1.2.1 Гауссовский шум

Модель гауссовского шума является одной из самых популярных моделей шума в задачах шумоподавления, так как он описывает естественные причины появления шума. Основные источники гауссовского шума в цифровых изображениях возникают во время получения, например шум датчика, вызванный плохим освещением или высокой температурой. Гауссовский шум является аддитивным, поэтому процесс искажения цифрового изображения можно описать следующей формулой.

$$y = x + n \quad (3)$$

где:

- $y$  - зашумленное изображение
- $x$  - исходное изображение
- $n$  - случайная величина имеющая гауссовское распределение.

Гауссовский шум можно описать плотностью вероятности имеющий вид:

$$P(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \quad (4)$$

где

- $P(z)$  - плотность вероятности
- $z$  - равномерно распределенная случайная величина
- $\sigma$  - среднеквадратичное отклонение
- $\mu$  - среднее значение

### 1.2.2 Пуассоновский шум

Немаловажным на практике является также пуассоновский шум. Он возникает из-за того, что количество фотонов, которые детектируется сенсором не всегда равное количество, даже при одинаковых условиях съемки. Количество фотонов, которые будут задетектированы описываются пуассоновским распределением. Рассмотрим выражения плотности вероятности для него.

$$P(X = k) = \frac{N^k}{k!} e^{-N} \quad (5)$$

где:

- $P(X = k)$  - плотность вероятности
- $X$  - случайная количество фотонов
- $k$  - количество фотонов, которые могут быть детектированы
- $N$  - реальное количество фотонов



## 1.3 Простейшие алгоритмы

Для демонстрации работы фильтров будет использовано следующее изображение.



Всякий раз мы будем на него накладывать аддитивный гауссовский шум со среднеквадратичным отклонением равным 0.05.



C:/Users/nikit/PycharmProjects/diploma/doc/img/exNoise

### 1.3.1 Вох фильтр

Первые фильтры были линейными, они были основаны на идее, что пиксели в некоторой малой окрестности имеют примерно одинаковые значения интенсивности. Поэтому если представлять каждый пиксель в виде суммы пикселей в окрестности, то это поможет избавиться от шума. Такой фильтр называется "Вох"фильтром. "Вох"фильтра задается квадратной матрицей с радиусом  $r$ , где каждый элемент матрицы равен  $\frac{1}{r^2}$ .

$\frac{1}{3^2}$	$\frac{1}{3^2}$	$\frac{1}{3^2}$
$\frac{1}{3^2}$	$\frac{1}{3^2}$	$\frac{1}{3^2}$
$\frac{1}{3^2}$	$\frac{1}{3^2}$	$\frac{1}{3^2}$

Рис. 1: Пример ядра "box"фильтра с радиусом 3

Ниже представлен пример работы фильтра.

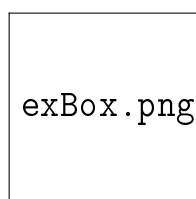


Рис. 2: Результат применения "box"фильтра с радиусом 5

Как можно видеть, результат работы "box"фильтра имеет артефакты в виде горизонтальных и вертикальных линий. Это является причиной почему данный фильтр не используют на практике.

### 1.3.2 Гауссовский фильтр

Улучшением идеи "Box"фильтра стал гауссов фильтр. В отличии от ядра "box"фильтра, значения ядра гауссовского фильтра вычисляются с помощью функции гаусса от двух переменных :

$$G(x, y) = \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (6)$$

где

- $x, y$  - координаты ядра
- $\sigma$  среднеквадратичное отклонение

Параметр  $\sigma$  обозначает насколько сильно будет размыто изображение, соответственно от  $\sigma$  зависит и радиус ядра фильтра. Т.е.  $r = 3\sigma$ . Ниже представлены графики одномерной функции гаусса с различными значениями  $\sigma$ .

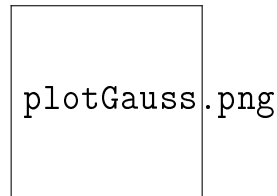
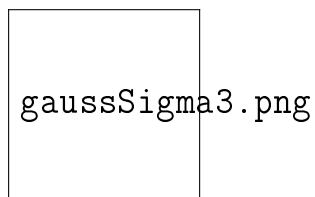
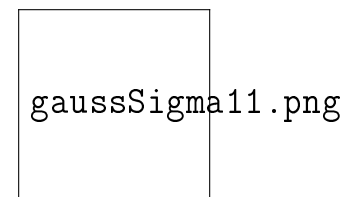


Рис. 3: Графики функции гаусса с параметром  $\sigma = 3, 5, 11$

Применим фильтр гаусса для рис. 1.3 с различными параметрами  $\sigma$ .



а)



б)

Рис. 4: Зависимость сигнала от шума для данных.

**1.3.2.1 Достоинства и недостатки "box" фильтра и фильтра гаусса** Преимуществом первых фильтров является простая реализация и быстрая скорость работы. Так как процесс шумоподавления, можно представить в виде поэлементного умножения спектрального образа шума на спектральный образ изображения. Но они также обладают одним существенным недостатком. Данные фильтры размывают края, это может негативно сказываться на алгоритмах компьютерного зрения.

### 1.3.3 Медианный фильтр

Избавиться от указанного выше недостатка попытался медианный фильтр. Он основан на той идее, что пиксели в некоторой малой окрестности имеют приблизительно равную интенсивность, а шум, соответственно сильно отличается. Поэтому для пикселя, для которого вычисляется новое значение, берутся пиксели в

некоторой окрестности. Как правило, это квадрат с радиусом  $r$ . Данные пиксели сортируются по возрастанию или убыванию и новым значением объявляется то, что находится в середине.

Продemonстрируем на примере работу фильтра.

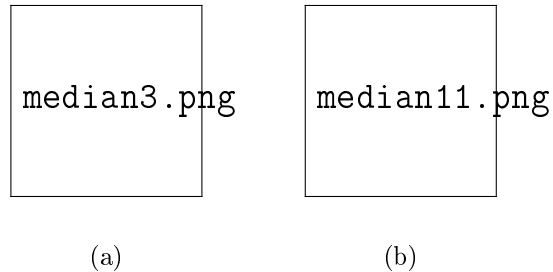


Рис. 5: Результат работы медианного фильтра: (a) с радиусом 3; (b) с радиусом 11

**1.3.3.1 Достоинства и недостатки медианного фильтра** Данный алгоритм работает хорошо, только с импульсным шумом, что сильно ограничивает его область использования. Но главным недостатком фильтра является то, что изображение теряет своё визуальное качество, становится более "мультикшным".

## 1.4 Обзор современных методов шумоподавления

Шумоподавление - это активно развивающаяся область, количество методов увеличивается с каждым годом. Поэтому будет удобным, их классифицировать по некоторым признакам. Методы разделяются на то, в каком домене они используются: частотный, пространственный. К пространственным относятся следующие алгоритмы: Билатеральный, Guided, Non-Local Means, Markov Random Field, Total Variation. К частотным: BM3D. Так же разделяются на тип обработки, существуют алгоритмы, которые обрабатывают изображение по пикселям или целыми блоками. К методу по пиксельной обработке относят: Total Variation. К блоковым: BM3D, Билатеральный, Guided, Non-Local Means.

## 1.5 Билатеральный фильтр

### 1.5.1 Основная идея

Билатеральный фильтр является некоторым улучшением фильтра гаусса, с тем лишь дополнением, что влияние пикселей уменьшается не только с увеличением расстояния до центра ядра фильтра, но так с увеличением разности интенсивностей.

### 1.5.2 Описание

Нахождения нового значения для пикселя, можно описать следующей формулой.

$$x(u) = \frac{\sum_{p \in N(u)} W_c(\|p - u\|) W_s(|y(u) - y(p)|) y(p)}{\sum_{p \in N(u)} W_c(\|p - u\|) W_s(|y(u) - y(p)|)} \quad (7)$$

где

- $x(u)$  - новое значение пикселя
- $u$  - пиксель, для которого высчитывается новое значение
- $y(u)$  и  $y(p)$  - значения пикселей в исходном изображении

- $N(u)$  - окрестность пикселя  $u$
- $W_c$  - весовая функция расстояния с параметром  $\sigma_c = \exp(\frac{-x^2}{(2\sigma_c^2)})$
- $W_s$  - весовая функция цвета с параметром  $\sigma_s = \exp(\frac{-x^2}{(2\sigma_s^2)})$

Весовая функция подбирается в зависимости от природы шума. В данном случае выбрана функция Гаусса.

Сохранение краёв на изображении достигается за счёт весовой функции цвета. Если мы считаем новый цвет оказавшись на "темной стороне т.е. где пиксели принимают низкие значения интенсивности , то как раз "темные" пиксели будут вносить больший вклад, в то время как более "светлые" пиксели, практически не будут влиять на результат. Диапазон интенсивности, который будет влиять на итоговое значение задаётся параметром  $\sigma_s$ , при чем зависимость эта прямо пропорциональная. Параметр  $\sigma_c$  определяет радиус окрестности, которая будет влиять на итоговое значение. На практике окрестность пикселя, т.е. её радиус =  $2\sigma_c$ .

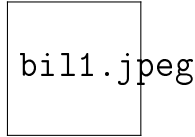
Рассчитаем сложность алгоритма. Для каждого пикселя, необходимо проверить все пиксели в определенном радиусе. В таком случае сложность алгоритма =  $O(nr^2)$ , где  $n$  - количество пикселей в изображении,  $r$  - радиус.

Одним из главных преимуществ данного фильтра, является его простота, а также сохранение границ. К сожалению, он так же обладает и рядом недостатков, среди которых:

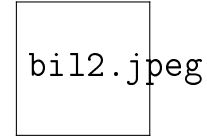
- Долгое время работы
- Артефакты на краях изображения

### 1.5.3 Результат работы фильтра

После применения билатерального фильтра с различными параметрами были получены следующие результаты.



а) Параметры 1



б) Параметры 2

## 1.6 Guided фильтр

### 1.6.1 Основная идея

Идея заключается в том, что бы найти линейную зависимость определенного блока пикселей исходного изображения с таким же блоком в управляемом изображении. Применяя данную задачу к проблеме шумоподавления, в качестве управляемого изображения используется зашумленное изображение.

### 1.6.2 Описание

Запишем линейную зависимость между пикселем на выходе фильтра и пикселем управляемого изображения в определенном окне следующим образом.

$$q_i = a_k I_i + b_k, \forall i \in \omega_k \quad (8)$$

где:

- $q_i$  - итоговое значение пикселя  $i$
- $I_i$  - значения пикселя  $i$  управляемого изображения
- $a_k, b_k$  - линейные коэффициенты
- $\omega_k$  - окно с центром в пикселе  $k$

Обычно в качестве окна используют квадрат с радиусом  $r$ . Для того, что бы определить линейные коэффициенты рассмотрим уравнение, которое минимизирует разницу между итоговым изображением  $q$  и входным изображением  $p$ . Введем



следующую целевую функцию.

$$E(a_k, b_k) = \sum_{i \in \omega_k} ((a_k I_i + b_k)^2 + \varepsilon a_k^2) \quad (9)$$

Здесь  $\varepsilon$  - регуляризационный параметр предотвращающий слишком большое значения  $a_k$ . Благодаря методу линейной регрессии получим следующие значения для линейных коэффициентов.

$$a_k = \frac{\frac{1}{|\omega_k|} \sum_{i \in \omega_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \quad (10)$$

$$b_k = \bar{p}_k - a_k \mu_k \quad (11)$$

где

- $\mu_k$  - среднее значение управляемого изображения  $I$  в окне  $\omega_k$
- $\sigma_k$  - среднеквадратичное отклонение изображения  $I$  в окне  $\omega_k$
- $|\omega|$  - количество пикселей в окне  $\omega_k$
- $\bar{p}_k$  - среднее значение исходного изображение  $I$  в окне  $\omega_k$

Теперь применим данную модель к целому изображению. Но пиксель  $i$  может находится одновременно в нескольких окнах  $\omega_k$ , поэтому итоговое значения  $q_i$  будет отличаться от локального. Простым решением будет взять среднее значение от всех  $\omega_k$  в изображении. В конечном итоге получим.

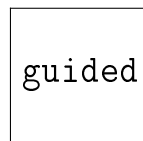
$$q_i = \frac{1}{|\omega_k|} \sum_{k: i \in \omega_k} (a_k I_i + b_k) = \bar{a}_k I_i + \bar{b}_k \quad (12)$$

Свойства сохранения краев можно объяснить следующим способом, возьмем вспомогательное изображение равное исходному т.е.  $I = p$ , в таком случае могут быть рассмотрены два крайних случая. В случае, когда окно будет гладким, т.е. значения пикселей будут мало отличаться друг от друга, в таком случае параметр  $a_k = 0$ , а  $b_k = \bar{p}_k$ . В противоположном случае,  $a_k = 1$ , а  $b_k = 0$ . Влияние параметра

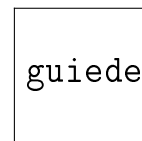
$\varepsilon$ , можно описать следующим образом. Если среднее отклонение  $\sigma^2$  меньше, чем  $\varepsilon$ , то область размывается, в ином случае она остается неизменной.

Главное особенностью данного фильтра является то, что его сложность зависит только от количества пикселей в изображении, т.е.  $O(n)$  где  $n$  - количество пикселей

### 1.6.3 Результат работы фильтра



а) Параметры 1



б) Параметры 2

## 1.7 Non-Local mean фильтр

### 1.7.1 Основная идея

Обычно, новое значение пикселя берется как некое усредненное значение его соседей. К таким фильтрам относятся: билатеральный фильтр, box фильтр и фильтр гаусса. Развитием этой идеи стал фильтр NL-mean. Суть его заключается в том, что бы искать похожие куски изображения на всей области изображения и брать некое среднее значение.

### 1.7.2 Описание

Идею приведенную выше можно математически описать в следующем виде.

$$x_i = \sum_j w(i, j) v(j) \quad (13)$$

где

- $x_i$  - итоговое значение пикселя.

- $w(i, j)$  - весовая функция, которая сравнивает две области изображения, с центром в пикселе  $i$  и  $j$
- $v(j)$  - интенсивность пикселя  $j$

Уравнение 13 указывает зависимость между значением итогового пикселя и всех пикселей исходного, зашумленного изображения.

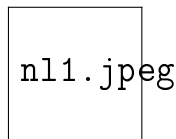
$$w(i, j) = \frac{\exp\left(-\frac{\|N(i)-N(j)\|_{2,a}^2}{h}\right)}{\sum_{j \in I} \exp\left(-\frac{\|N(i)-N(j)\|_{2,a}^2}{h}\right)} \quad (14)$$

где

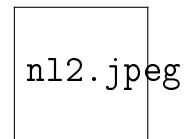
- $h$  - параметр указывает степень фильтрации, чем он меньше, тем менее похожие паттерны изображения будут влиять на итоговый пиксель
- $N(i)$  и  $N(j)$  - соседи пикселей  $i$  и  $j$
- $a$  - стандартное распределение гауссовского фильтра

Не трудно заметить, что сложность данного алгоритма  $O(n^2(r*2+1))$ , это приводит к значительным временным затратам даже для изображений малых размеров. Для уменьшения сложности вычисления, можно сделать следующие упрощения: можно заранее рассчитать все веса, так как область одного и того же пикселя может использоваться несколько раз, можно уменьшить окно поиска похожих патчей, т.е. теперь похожие патчи будут искажаться не во всем изображении, а в некотором окне. Так же можно сравнивать расстояния не для каждого пикселя, а через один, два и т.д. Тогда сложность будет  $= O(n * N * (r * 2 + 1))$ , где  $N$  размер большого окна.

### 1.7.3 Результат работы фильтра



а) Параметры 1



б) Параметры 2

## 1.8 BM3D

Фильтр BM3D можно считать более лучшей версией алгоритма Non-Local mean. В нём так же одним из ключевых моментов является усреднение значений посредством поиска похожих блоков. В алгоритме можно выделить два основных шага. Каждый шаг в свою очередь разбивается на 3 этапа. Первый этап называется группировка (*grouping*), в котором для каждого патча (часть изображения фиксированного размера), отбираются другие похожие патчи. Вторым этапом является совместная (*collaborative*) фильтрация. Третьим этапом является агрегация, где для каждого пикселя вычисляется новое значение, с учетом того, что данный пиксель мог находится в различных патчах.

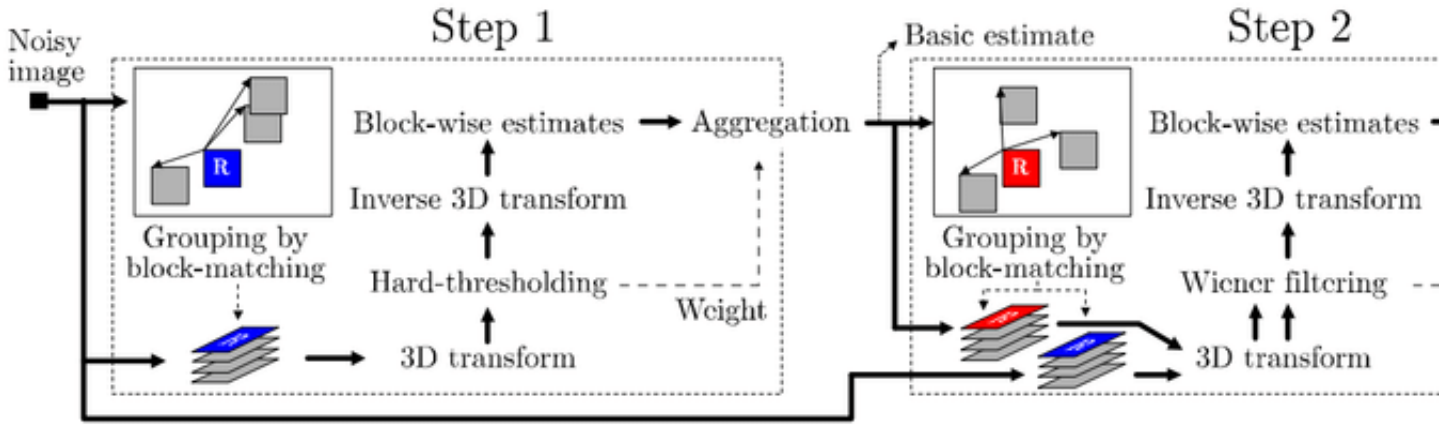


Рис. 6: Схема алгоритма BM3D

Далее патчем будет обозначаться часть исходного изображения с фиксированным размером.

### 1.8.1 Оценка дисперсии

Для корректной работы алгоритма необходимо правильно оценивать дисперсию. Для этого используется следующая метрика - MAD (median absolute deviation). Выглядит она следующим образом.

$$\sigma = k * MAD \quad (15)$$

где

- $\sigma$  - дисперсия
- $MAD$  - медианное абсолютное отклонение
- $k$  - константа, которая зависит от распределения. В случае нормального распределения  $k = 1.4826$

## 1.8.2 Первый шаг

### 1.8.2.1 Группировка

Всё изображение разбивается на патчи (блоки), которые имеют размером -  $N_{ht}^{Psize}$ . Далее для каждого патча, обозначим его  $P$ , производятся следующие действия. В окне размера -  $N_{ht}^W size$ , в котором центром является  $P$ , с шагом -  $N_{ht}^{Sstep}$  ищутся похожие блоки 16.

$$G_{ht}(P) = \{Q | d_{ht}(Q, P) \leq h_{ht}\} \quad (16)$$

где

- $G_{ht}(P)$  - множество блоков схожих с  $P$
- $Q$  - один из блоков в окне
- $d_{ht}(Q, P) = \frac{\|\gamma_{\lambda_{ht}^{2D}}(P) - \gamma_{\lambda_{ht}^{2D}}(Q)\|_2^2}{(N_{ht}^{Psize})^2}$
- $\gamma_{\lambda}(x) = \begin{cases} 0 & |x| \leq \lambda \\ x & |x| > \lambda \end{cases}$  - жесткая пороговая фильтрация
- $\lambda_{ht}^{2d}$  - для низких значений шума  $\sigma < 40$  обычно равняется нулю, подробнее [4]
- $\sigma^2$  - дисперсия шума
- $h_{ht}$  - порог, для того, что бы считать два блока похожими.

Максимальное количество подходящих блоков так же ограничивают, за это отвечает параметр -  $N_{ht}^{Pmax}$ . Найденные блоки объединяются в 3D-блок, далее обозначаемый  $G_{ht}^{3D}(P)$ , в котором блоки располагаются по мере уменьшения схожести.

### 1.8.2.2 Совместная фильтрация

К  $G_{ht}^{3D}(P)$  применяется 3D преобразование -  $r_{ht}^{3D}$ , трансформирующие его в частотную область, после применяется жесткая пороговая фильтрация  $\gamma_{\lambda_{ht}^{3D}}$ . После данных операций происходит обратное 3D преобразование -  $r_{ht}^{3D^{-1}}$ . Данные действия описываются следующей формулой.

$$G_{ht}^{3D'}(P) = r_{ht}^{3D^{-1}}(\gamma_{\lambda_{ht}^{3D}}(r_{ht}^{3D}(G_{ht}^{3D}(P)))) \quad (17)$$

### 1.8.2.3 Агрегация

Теперь для каждого пикселя в изображении, необходимо рассчитать новое значение. При этом нужно учесть, что пиксель мог находиться в нескольких  $G_{ht}^{3D'}$ . Итоговая формула следующая.

$$y^{basic}(x) = \frac{\sum_P w_p^{hard} \sum_{Q \in G_{ht}^{3D'}(P)} \chi_Q(x) u_{P,Q}(x)}{\sum_P w_p^{ht} \sum_{Q \in G_{ht}^{3D'}(P)} \chi_Q(x)} \quad (18)$$

где

- $y^{basic}(x)$  - значение пикселя после первого шага
- $w_p^{ht} = \begin{cases} (N_p^{hard})^{-1} & N_p^{hard} \geq 1 \\ 1 & N_p^{hard} < 1 \end{cases}$
- $N_p^{hard}$  - количество ненулевых пикселей после  $\gamma_{\lambda_{ht}^{3D}}(r_{ht}^{3D}(G_{3D}(P)))$
- $\chi_Q(x)$  - равняется 1, если  $x \in Q$

### 1.8.3 Второй шаг

На основе  $y^{basic}$  можно получить более лучшее шумоподавления, применяя фильтр Винера. Патч, для которого произведены дальнейшие действия обозначим  $P^{basic}$

#### 1.8.3.1 Группировка

Будем считать, что количество шума после первого шага сведенно к минимуму. Определим формулу, для поиска подходящих блоков на втором шаге.

$$G_{wie}(P^{basic}) = \{Q | d_{wie}(Q, P^{basic}) \leq h_{wie}\} \quad (19)$$

где

- $G_{wie}(P^{basic})$  - множество блоков схожих с  $P^{basic}$
- $Q$  - один из блоков в окне
- $d_{wie}(Q, P^{basic}) = \frac{\|P^{basic} - Q\|_2^2}{(N_{ht}^{Psize})^2}$
- $h_{wie}$  - порог, для того, что бы считать два блока похожими.

**1.8.3.2 Совместная фильтрация** Объединим блоки найденные на этапе группировка второго шага в  $G_{wie}^{3D}(P^{basic})$ . Затем применим 3D преобразование -  $r_{wie}^{3D}$ , как в первом шаге. Применим фильтр Винера: поэлементно умножим  $G_{wie}^{3D}(P^{basic})$  на коэффициент сжатия Винера -  $w_P$ . После, применим обратное 3D преобразование -  $r_{ht}^{3D^{-1}}$ . Опишем эти действия следующей формулой.

$$G_{wie}^{3D'}(P^{basic}) = r_{wie}^{3D^{-1}}(w_P \cdot r_{wie}^{3D}(G_{wie}^{3D}(P))) \quad (20)$$

где

- $w_P = \frac{|r_{wie}^{3D}(G_{wie}^{3D}(P^{basic}))|^2}{|r_{wie}^{3D}(G_{wie}^{3D}(P^{basic}))|^2 + \sigma}$

### 1.8.3.3 Агрегация

Агрегация на втором шаге проводится по следующей формуле.

$$y(x) = \frac{\sum_P w_p^{wie} \sum_{Q \in G_{wie}^{3D'}(P)} \chi_Q(x) u_{P,Q}(x)}{\sum_P w_p^{wie} \sum_{Q \in G_{wie}^{3D'}(P)} \chi_Q(x)} \quad (21)$$

где

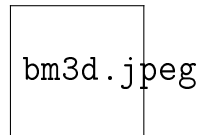
- $y(x)$  - итоговое значение пикселя
- $w_p^{wie} = \|w_P\|_2^{-2}$

### 1.8.4 Оптимальные параметры

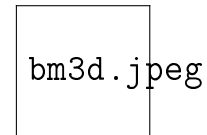
Согласно [5], оптимальными параметрами будут следующие:

Обозначение параметра	$\sigma \leq 40$	$\sigma > 40$
$N_{ht}^{Pmax}$	16	16
$N_{wie}^{Pmax}$	32	32
$N_{ht}^{Psize}$	8	8
$N_{wie}^{Psize}$	8	8
$\lambda_{ht}$	2.7	2.7
$h_{ht}$	2500	5000
$h_{wie}$	400	3500

### 1.8.5 Результат работы фильтра



а) Параметры 1



б) Параметры 2



## 1.9 Total Variation фильтр

### 1.9.1 Основаная идея

Для описания основной идеи потребуются определить вариацию функции (total variation) - показывает насколько сильно изменился сигнал между своими значениями. Применяя это описания для одномерного дискретного конечного сигнала  $x$  длиной  $N$  получим.

$$TV(x) = \sum_{n=2}^N |x(n) - x(n-2)| \quad (22)$$

Так же вариацию функции удобно описать в матричном виде.

$$TV(x) = \| xD \|_1 \quad (23)$$

где  $\| \cdot \|_1$  -  $\ell_1$  норма а,

$$D = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix} \quad (24)$$

матрица размером  $(N-1) \times N$  Предполагается, что на бесшумном изображении вариация функции меньше, чем на зашумленном, поэтому данную функцию можно использовать в качестве регуляризационного члена целевой функции, которая будет определена далее.

### 1.9.2 Описание

Необходимо найти неискаженное шумом изображение  $x$  с помощью зашумленного изображения  $y$ . Для этого введем целевую функцию, где регуляризационным членом будет выражение 23.

$$J(x) = \lambda \| Dx \|_1 + \| x - y \|_2^2 \quad (25)$$

где:

- $x$  - итоговое изображение, представленное в виде матрицы
- $D$  - трехмерная матрица определенная в 24
- $\lambda$  - регуляризационный параметр
- $y$  - зашумленное изображение, представленное в виде матрицы

Соответственно оптимальное значение можно обозначить следующим образом:

$$J_* = \min_x \|y - x\|_2^2 + \lambda \|Dx\|_1 \quad (26)$$

Нахождения оптимального значения является сложной задачей, из-за того что  $\ell_1$  норма не дифференцируема. Рассмотрим задачу как двойственную. Для этого представим  $\ell_1$  норму в следующей форме.

$$\|x\|_1 = \max_{|z| \leq 1} z^t x \quad (27)$$

Таким образом мы получим:

$$J_x = \min_x \max_{|z| \leq 1} \|y - x\|_2^2 + \lambda z^t Dx \quad (28)$$

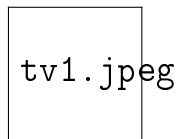
Описание вывода решения двойственной задачи 28 будет опущено, так как оно не входит в рамки данной ВКР. Результатом является итерационный алгоритм, который выглядит следующим образом:

$$x^{(i+1)} = y - D^t z^{(i)}$$

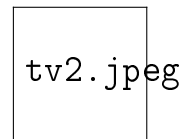
$$z^{(i+1)} = \text{clip}(z^{(i)} + \frac{1}{\alpha} D x^{(i+1)}, \frac{\lambda}{2})$$

for  $i \geq 0$  with  $z^{(0)} = 0$  and  $a \geq \max \text{eig}(A, A')$

### 1.9.3 Результат работы фильтра



а) Параметры 1



б) Параметры 2

## 1.10 Заключение

В данном разделе была поставлена задача шумоподавления, были рассмотрены самые простые методы обработки изображений и выявлены их недостатки. Также были классифицированы некоторые продвинутое методы шумоподавления. Были описаны основные идеи каждого фильтра и их математическое описание.

## 2 Критерии оценки эффективности алгоритмов

### 2.1 PSNR

Для оценки степени искажения изображений, используются различные методы. Самым популярным является PSNR(pick signal-noise-to-noise ratio). Он обозначает соотношение между максимумом возможного значения изображения и мощностью шума, который искажает данное изображение. Значения лежат в диапазоне от  $(0, \infty)$ , чем больше PSNR, тем изображение считается близки к оригиналу.

$$PSNR = 10 \log_{10} \left( \frac{MAX_I^2}{MSE} \right) \quad (29)$$

где

- $MAX_I$  - максимально возможное значения пикселя
- $MSE = \frac{1}{nm} \sum_{i=0}^n \sum_{j=0}^m (x_{i,j} - y_{i,j})^2$  - средний квадрат ошибки между оригинальным изображением  $x$  и искаженным  $y$ .
- $m, n$  - высота и ширина изображения

PSNR имеет как ряд преимуществ так и некоторые недостатки. К преимуществам относят быстроту и легкость вычисления. К недостаткам: для вычисления необходимо иметь исходную и преобразованную картинку, при этом не может быть точно известно, что оригинальная картинка не имела искажений, так же в ряде случаев PSNR плохо коррелирует с субъективными мерами оценки качества. Т.е. высокие значения PSNR не всегда обеспечиваются лучшим качеством картинки с визуальной точки зрения.

### 2.2 SSIM

Из-за несовершенств метрики PSNR разрабатывались все новые и новые методики. Одной из которых является SSIM (Structural SIMilarity). Отличительной

особенностью метода, является то, что метод 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.

$$SSIM = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{x,y} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (30)$$

где

- $\mu_x$  - среднее  $x$
- $\mu_y$  - среднее  $y$
- $\sigma_x^2$  - дисперсия  $x$
- $\sigma_y^2$  - дисперсия  $y$
- $\sigma_{x,y}^2$  - ковариация  $x$  и  $y$
- $c_1 = (k_1L)^2, c_2 = (k_2L)^2$  - две переменных
- $L$  - динамический диапазон (максимальное значение пикселя)
- $k_1 = 0.01, k_2 = 0.03$  - константы

Формула 30 указана только для яркостной компоненты. Значения находятся в отрезке  $(-1, 1)$ , если значение  $+1$ , то изображение наиболее похоже. При полном несоответствии  $-1$ . Так же данная метрика зависит от размера окна, обычно размер берется  $8 \times 8$ .

### 2.2.1 Выбор оптимального алгоритма

На основе оценок эффективности алгоритмов приведенных выше был предложен следующий метод оценки оптимального алгоритма. Для каждой категории

изображений производятся испытания всех реализованных алгоритмов с различными параметрами. Алгоритмы, которые показали максимальные значения SSIM и PSNR для большего количества изображений объявляются оптимальными.

## 2.3 Заключение

Развитие метрик оценивания схожести двух изображений продолжается и по сей день. Охватить их в полном объеме не получится, поэтому для дипломной работы были выбраны наиболее популярные метрики, хоть они и обладают некоторым недостатком. А именно, для них необходимо иметь не зашумленное изображения. Что в целях ВКР не является существенным. Так же предложен собственный метод выбора оптимального алгоритма, для определенной категории изображений.

## 3 Исследования

### 3.1 Тестовое множество

#### 3.1.1 Изображения

Для исследования были взяты изображения из открытой базы данных ImageNet, а так же из личной коллекции. Была произведена следующая классификация:

- Изображения людей
- Изображения архитектуры
- Изображения полученные при недостаточной освещенности

Данная классификация обусловлена распространенностью данных классов изображений и прикладных задачах в области обработки изображений. Фотографии людей используются как в повседневной жизни, так и во многих алгоритмах компьютерного зрения, таких как например распознавание лиц.

Не менее распространены так же и изображения полученные при недостаточной освещенности, что приводит к низко контрастному изображению. Это мешает их анализу. Так же данные это актуально и для обычных пользователей фото и видеокамер, устройство которых все еще не позволяет достичь приемлимых результатов при съемке ночью.

Последняя категория была выбрана в качестве примера, где некоторые части изображения похожи друг на друга. Это может быть полезно в таких областях связанных с медициной, электроникой и астрономией. Где изображения, необходимые для последующей обработки имеют подобные свойства.

Ниже приведены примеры изображений из каждой категории.

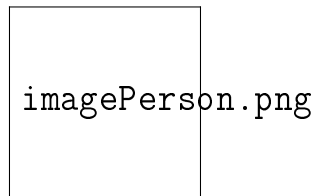


Рис. 7: Изобр. категории человек.

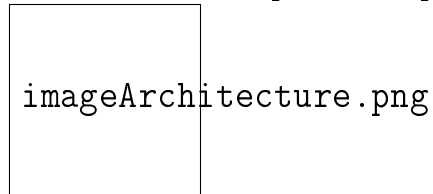


Рис. 8: Изобр. категории архитектура.

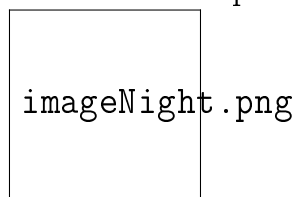


Рис. 9: Изобр. категории низкая освещенность.

### 3.1.2 Видео

Данные необходимые для исследования видео последовательность были взяты из открытой базы данных YouTube-8М.

Ниже приведены кадры из видео каждой категории.



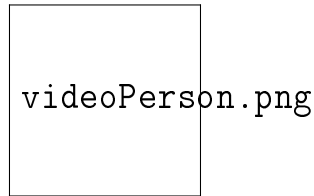


Рис. 10: Видео категории человек.

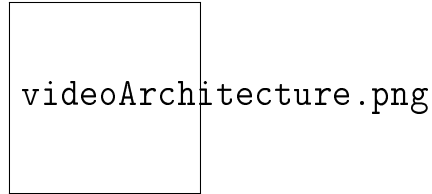


Рис. 11: Видео категории архитектура.

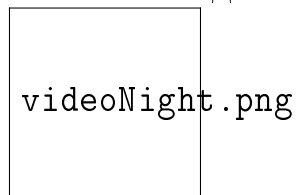


Рис. 12: Видео категории низкая освещенность.

## 3.2 Схема программы

Работу реализованной программы можно описать следующим образом. На вход программы подаются данные (видео или изображение). Далее накладывается гауссовский шум. После проводится шумоподавления различными алгоритмами с различными параметрами. В итоге выбирается алгоритм, которые показал наибольшие показатели PSNR и SSIM. Для каждой категории берутся максимальные и средние значения значения PSNR, SSIM.

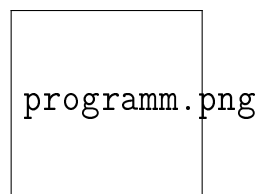


Рис. 13: Схема итоговой программы

### 3.3 Результаты исследования

Исследование разбито на две части. В первой части для каждого алгоритма подбираются оптимальные параметры, которые дают максимальное значение PSNR или SSIM для определенного типа изображения или видео. Исходя уже из полученных результатов будут сравниваться между собой все алгоритмы.

## A Billateral

# Список литературы

## Список литературы

- [1] Д. Калинкина, Ватолин Д. Проблема подавления шума на изображениях и видео и различные подходы к ее решению. 2005. <http://masters.donntu.org/2013/fknt/lashchenko/library/ni.pdf> дата обращения 02.05.2019.
- [2] Rohankar, Jayant.
- [3] К. He, J. Sun, X. Tang.
- [4] K. Dabov A. Foi V. Katkovnik, Egiazarian K. Image denoising by sparse 3D transform-domain collaborative filtering // IEEE Trans. Image Process. 2007. Т. 16. С. 2080–2095.
- [5] Lebrun Marc. An Analysis and Implementation of the BM3D Image Denoising Method // Image Processing On Line. 2012. Т. 2. С. 175–213.