

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение высшего
образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет «Вычислительная техника» (наименование)	Кафедра «Системы автоматизированного проектирования» (наименование)
Направление подготовки	09.03.01 «Информатика и вычислительная техника» (код и наименование)
Профиль	«Системы автоматизированного проектирования» (наименование)

БАКАЛАВРСКАЯ РАБОТА
на тему

Разработка компонентов системы wavelet-преобразования изображений

Студент Алимов 19.05.17 Никитин А.С.
(подпись, дата) (фамилия, инициалы)

Руководитель Бурз 1.06.17 Бурзун И.И.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр Уф. каф. САПР ПГУ Бурз 21.05.17 Бурзун А.С.
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой Бурз Бурзун И.И.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____
(подпись) (фамилия, инициалы)

Пенза 2017

РЕФЕРАТ

Пояснительная записка содержит __ с., __ рис., 12 источников литературы, 12 приложений.

WAVELET, WAVELET-ПРЕОБРАЗОВАНИЕ, ФИЛЬТРАЦИЯ ИЗОБРАЖЕНИЙ, ТРЕШОЛДИНГ, ЛИНЕЙНЫЙ ФИЛЬТР, УСРЕДНЯЮЩИЙ ФИЛЬТР, ФИЛЬТР ГАУССА, МЕДИАННЫЙ ФИЛЬТР, АДАПТИВНЫЙ ФИЛЬТР ВИНЕРА, ПРОГРАММИРОВАНИЕ, ООП, C#, VISUAL STUDIO, СИСТЕМА КОМПЬЮТЕРНОЙ МАТЕМАТИКИ, MATLAB.

Объектом разработки являются компоненты системы wavelet-преобразования изображений.

Цель работы – разработка компонентов системы wavelet-преобразования изображений, обеспечивающих загрузку изображения, выбор базового вейвлета, определение уровня фильтрации, фильтрацию изображения, визуализацию результатов и сохранение результатов.

В процессе работы выполнены следующие исследования и разработки: рассмотрена теория wavelet-преобразований, линейный, медианный и адаптивный фильтры, виды шумов и трешолдинг, разработано программное обеспечение компонентов системы, сопоставлены результаты работы рассмотренных методов.

Результатом данной работы является система wavelet-преобразования изображений, написанная на языке C# в среде Microsoft Visual Studio 2012 с использованием системы компьютерной математики Matlab.

					ВКР 090301.10. 17 81 01			
Изм.	Лист	№ докум	Подп.	Дата	<i>Разработка компонентов системы wavelet-преобразования изображения. Пояснительная записка.</i>	Лит	Лист	Листов
Разраб		Никонов А.С.						
Провер.		Бурукина И.П.					3	
Т.конт.						13BB2		
Н.конт.		Бождай А.С.						
Утв.		Бершадский А.М.						

Содержание

ВВЕДЕНИЕ	6
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	8
1.1 Постановка задачи.....	8
1.2 Основы теории wavelet-преобразования.....	10
1.3 Шумоподавление изображений.....	16
1.4 Обзор аналогичных методов.....	21
1.4.1 Преобразование Фурье.....	21
1.4.2 Линейный фильтр.....	23
1.4.3 Медианный фильтр	25
1.4.4 Адаптивный фильтр	26
1.4.5 Таблица сравнения аналогов	27
1.5 Обоснование необходимости разработки	28
1.6 Формализованная постановка задачи.....	30
1.6.1 Требования к функциональным характеристикам	30
1.6.2 Требования к надежности	31
1.6.3 Требования к составу и параметрам технических средств	32
1.6.4 Требования к информационной и программной совместимости	33
2 ОБОСНОВАНИЕ СРЕДСТВ РАЗРАБОТКИ.....	34
2.1 Выбор операционной системы	34
2.2 Выбор системы компьютерной математики	36
2.3 Выбор языка программирования и среды разработки	38
3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	40
3.1 Входные и выходные данные	40
3.2 Разработка класса wavelet-преобразования изображения.....	41

3.3 Разработка компонентов наложения шума	47
3.4 Разработка фильтров	49
3.5 Разработка главного окна	53
4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ.....	55
4.1 Наложение шума на изображения	55
4.2 Фильтрация изображений при помощи wavelet-преобразований	56
4.3 Использование фильтров	58
5 АНАЛИЗ РЕЗУЛЬТАТОВ	59
5.1 Гауссов шум.....	60
5.2 Шум «соль и перец».....	59
5.3 Мультипликативный шум.....	62
ЗАКЛЮЧЕНИЕ	63
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	66
Приложение А	68
Приложение Б.....	72
Приложение В	76
Приложение Г	79
Приложение Д	81
Приложение Е.....	81
Приложение Ж	83
Приложение З	85
Приложение И	92
Приложение К	96
Приложение Л	100
Приложение М	100

ВВЕДЕНИЕ

С развитием компьютерных технологий человечество научилось получать, обрабатывать, передавать и хранить информацию в различном виде. Это может быть текстовая или числовая информация, графическая или изобразительная, звуковая или видеоинформация. В качестве носителей могут использоваться различные бумажные материалы, магнитные материалы, такие как магнитные ленты или жесткие диски, оптические электронные носители, например, CD- или DVD-диски, или полупроводниковые носители, например, флеш-память.

Все эти носители позволяют в той или иной степени хранить различные виды информации. Таким образом, текстовое или графическое содержимое может быть зафиксировано на бумажном носителе, а звуковая информация может косвенно храниться в виде нот. Электронные носители позволяют хранить информацию в цифровой форме.

Графическая информация имеет большое значение в современном мире. Даже в первобытном обществе люди создавали наскальные рисунки, стремясь передать какую-то информацию. В современном мире графическое отображение не утратило своей актуальности. Люди используют графику в различных сферах: в искусстве, науке и др.

Технический прогресс преподнес возможность создавать, хранить и обрабатывать графическую информацию не только в бумажном, но и в электронном виде. Люди получили возможность создавать и обрабатывать различные изображения. Сначала фотографии хранились в печатном виде. Первоначально они были черно-белыми. Благодаря научно-техническому прогрессу, в настоящее время имеется возможность создавать высококачественные изображения и хранить их не только на бумажных носителях, но и в электронной форме на различных цифровых носителях.

При работе с изображениями возникает потребность в их обработке. Это связано с тем, что изображения могут быть повреждены, например, зашумлены. Причиной этого может быть неисправность фотоаппаратуры, неподходящие условия съемки, повреждения носителя информации или ошибки при передаче информации. Для улучшения качества изображений создаются специальные программные средства.

В настоящий момент существуют различные технологии разработки программных средств. В их основе лежат такие подходы, как структурный или объектно-ориентированный. Они описывают разработку программ методами нисходящего программирования (от общего к частному) или восходящего программирования (от частного к общему) соответственно. Для разработки программ имеются разные языки программирования. Также разработаны интегрированные среды разработки (integrated development environment), предоставляющие широкие возможности программистам.

В основе программ лежат определенные методы или алгоритмы. Именно с этих вопросов начинается разработка программ. Таким образом, для повышения качества изображения, в частности для устранения шумов, применимы определенные подходы, которые будут рассмотрены в следующих пунктах.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Постановка задачи

В рамках выпускной квалификационной работы необходимо разработать компоненты системы wavelet-преобразования изображения.

Система предназначена для повышения качества изображений за счет применения wavelet-преобразования.

Основные функции системы:

1. Загрузка изображения.
2. Выбор базового вейвлета.
3. Определение уровня фильтрации.
4. Визуализация результатов.
5. Сохранение результатов.

Таким образом, система должна выполнять очистку (фильтрацию) зашумленного изображения при помощи wavelet-преобразования.

Следует предусмотреть возможность наложения шумов различного вида и в различной степени на изображение, для оценки результатов фильтрации в зависимости от выбранного базового вейвлета, уровня преобразования и параметров очистки шума.

Требуется реализовать возможность загрузки в программу исходного изображения. Оно может быть предварительно зашумлено в соответствующем компоненте или быть представлено в исходном виде.

Необходимо предусмотреть выбор базового вейвлета в программе. Следовательно, следует определить набор используемых вейвлетов в зависимости от их свойств.

Помимо этого, следует предусмотреть возможность определения уровня фильтрации, т.е. определение уровня wavelet-преобразования.

Необходимо определить способ устранения шумов на изображении и выделить его входные параметры. В программе следует реализовать конкретный способ и возможность ввода исходных параметров для него.

Заключительным этапом очистки изображения является сохранение полученного изображения. Следовательно, необходимо предусмотреть в программе процедуры сохранения результатов работы, т.е. очищенного изображения.

В данной выпускной квалификационной работе необходимо сделать выводы о результатах работы программы, определить зависимость результатов от выбранных входных параметров (вейвлета, уровня преобразования, параметров шумоподавления), от различных шумов на изображении и степени зашумленности.

Для более полной оценки результатов wavelet-преобразования надлежит рассмотреть и реализовать аналогичные фильтры, в основе которых лежат определенные методы или алгоритмы устранения шумов на изображении. Следовательно, стоит сравнить полученные результаты в зависимости от наложенного шума и его степени и выбранного способа фильтрации с соответствующими параметрами.

Для разработки программы необходимо определить технологии разработки, выбрать язык программирования, методологию программирования и спроектировать архитектуру разрабатываемых компонентов системы.

1.2 Основы теории wavelet-преобразования

Wavelet – это математическая функция в виде короткой волны (всплеска), применяемая для обработки сигналов или изображений. Термин «wavelet» дословно означает «маленькая волна». Слово «волна» говорит о том, что происходит периодическое изменение. Данный термин был введен Хааром и далее проработан Добеши и Морле.

Применение вейвлетов весьма разнообразно. Они используются не только при обработке или очистке сигналов и изображений, но и в задачах распознавания образов (pattern recognition), в геофизике, радиотехнике, медицине и для сжатия больших объёмов информации (в том числе видеоинформации).

С использованием вейвлетов входной сигнал (или изображение) представляется совокупностью волновых пакетов, образованных на основе определенной функции[6]. Это называется wavelet-анализом.

Вейвлеты имеют следующие свойства[6]:

1. Имеют вид коротких волн.
2. Обладают возможностью сдвига по оси времени.
3. Способны к масштабированию.

Существуют различные виды вейвлетов. Они будут описаны позже. Wavelet выбирается в зависимости от многих параметров, в том числе от требуемых результатов.

Математическим аппаратом, использующим вейвлеты, называется wavelet-преобразование. Wavelet-преобразование представляет собой скалярное произведение вейвлета и анализируемого сигнала (например, изображения)[1].

Wavelet-преобразование – это разделение имеющегося сигнала или изображения на составляющие с использованием вейвлетов. Wavelet-преобразование бывает непрерывным и дискретным. Непрерывное wavelet-преобразование реализуется в виде интегральной функции. Оно сложно и трудоемко, поэтому в рамках данной выпускной квалификационной работы

будет использоваться дискретное wavelet-преобразование. В результате дискретного wavelet-преобразования получается две составляющие: аппроксимирующая и детализирующая. Аппроксимирующая низкочастотная компонента несет в себе основную информацию об изображении, детализирующая высокочастотная – детали изображения. Другими словами, дискретное wavelet-преобразование позволяет получить аппроксимирующие и детализирующие коэффициенты, которые описывают сглаженный сигнал и его колебания соответственно[8].

Ниже на рисунке 1 представлена классификация вейвлетов[8].



Рисунок 1 – Классификация вейвлетов

К предвейвлетам относятся гауссовы вейвлеты, Морле, мексиканская шляпа. Они являются симметричными. Не являются ортогональными.

Регулярные и дискретные вейвлеты Мейера являются ортогональными и симметричными. Возможно дискретное преобразование.

К ортогональным вейвлетам с компактным носителем относятся Добеши2, Добеши4, симплет2, симплет4, койфлет2, койфлет4. Они обеспечивают ортогональный анализ, дискретное преобразование. Вейвлеты Добеши являются несимметричными. Остальные вейвлеты почти симметричны[8].

Биортогональные пары вейвлетов с компактным носителем – это В-сплайновый биортогональные вейвлеты.

Комплексные вейвлеты – комплексные гауссовы вейвлеты, комплексные вейвлеты Морле.

Wavelet-преобразование позволяет работать со стационарными и нестационарными сигналами [4]. Стационарный сигнал – это сигнал, частота

которого не меняется во времени. Нестационарный сигнал – это сигнал с изменяющейся частотой.

Алгоритм многоуровневого wavelet-преобразования заключается в следующем. По набору значений сигнала вычисляются аппроксимирующие и детализирующие коэффициенты. Они получаются применением высокочастотного фильтра для детализирующих коэффициентов и низкочастотного фильтра для аппроксимирующих коэффициентов. Далее выполняется операция децимации. Она заключается в «прореживании» сигнала. На следующей итерации алгоритма аппроксимирующие коэффициенты подвергаются аналогичному воздействию. Такой подход получил название алгоритма Маллата.

Данная схема применима не только для сигналов, но и для изображений.

Схематично алгоритм многоуровневого wavelet-преобразования представлен на рисунке 2.

На рисунке 2 имеются следующие обозначения:

- S – сигнал или изображение.
- Low filter – низкочастотный фильтр.
- High filter – высокочастотный фильтр.
- $\downarrow 2$ – операция децимации.
- Ca – аппроксимирующие коэффициенты.
- Cd – детализирующие коэффициенты.

Рисунок 2 наглядно демонстрирует работу многоуровневого wavelet-преобразования. В нем последовательно на каждом уровне формируются аппроксимирующие коэффициенты при помощи низкочастотного фильтра и децимации. Количество получаемых наборов аппроксимирующих коэффициентов определяется заданным уровнем wavelet-преобразования. Детализирующие коэффициенты при этом отбрасываются. Существуют пакетные вейвлеты. При их использовании детализирующие коэффициенты не отбрасываются, а подвергаются дальнейшей обработке. Такой подход

неприменим для очистки изображений, т.к. в детализирующих коэффициентах может содержаться шумовая компонента.

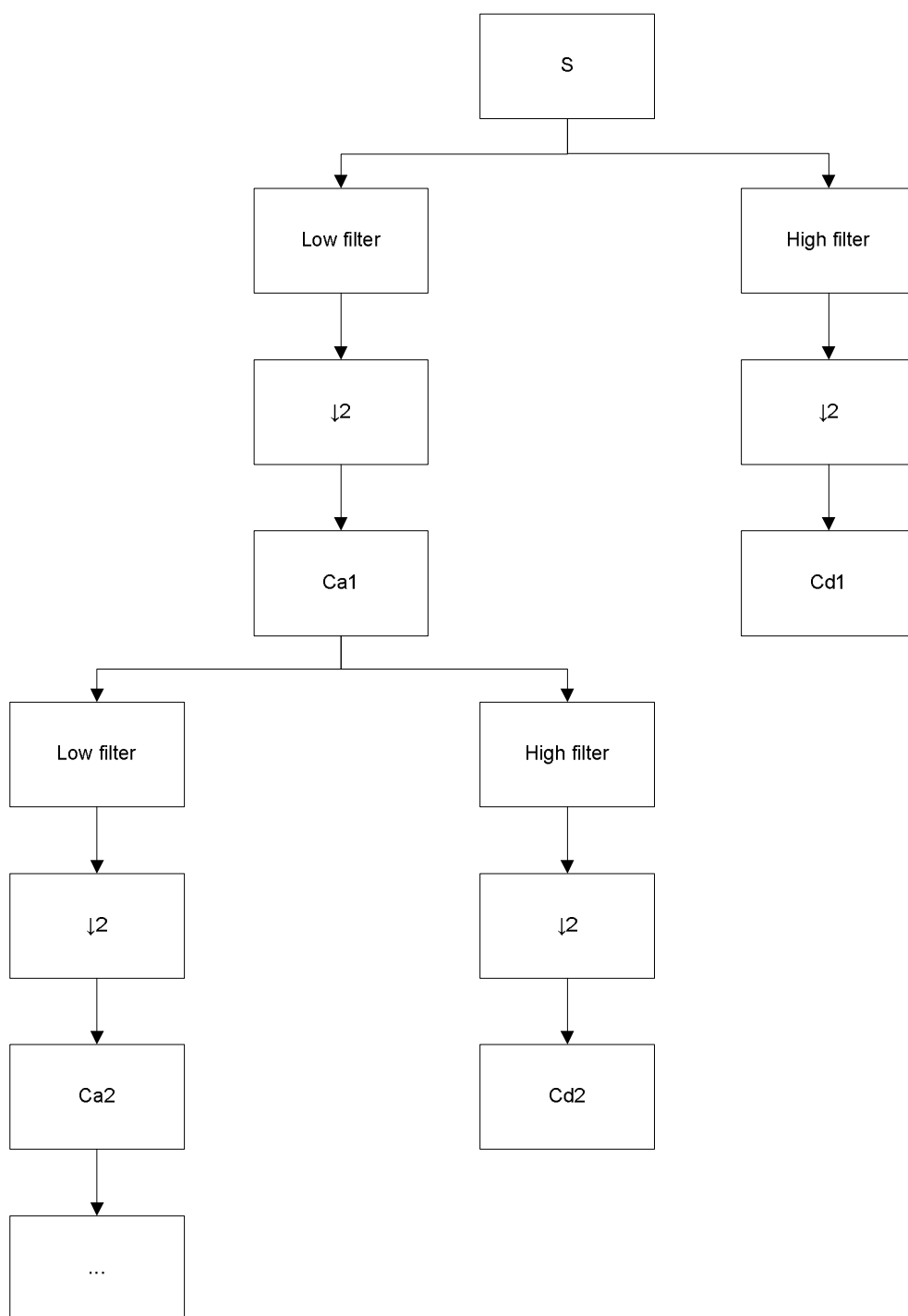


Рисунок 2 – Схема многоуровневого wavelet-преобразования

Wavelet-преобразование позволяет реконструировать сигнал или изображение, то есть осуществлять обратное wavelet-преобразование. Это используется при очистке изображения от шумов, так как после выполнения многоуровневого преобразования необходимо восстановить изображение.

Для устранения шумов при помощи wavelet-преобразования базовый wavelet должен быть ортогональным и иметь возможность реконструкции сигнала[7]. Следовательно, исходя из ранее изложенной классификации вейвлетов, для разработки программы следует выбрать следующие вейвлеты: Хаара, Добеши2, Добеши4, симплет2, симплет4, койфлет2, койфлет4. Рассмотрим выбранные вейвлеты подробнее.

Вейвлет Хаара (Haar wavelet) является наиболее простым и одним из первых ортогональных вейвлетов. Вейвлет Хаара хорошо локализован в пространстве, но не очень хорошо в частотной области[6]. Недостатком вейвлета является его негладкость и несимметричность. На рисунке 3 представлен график вейвлета Хаара.

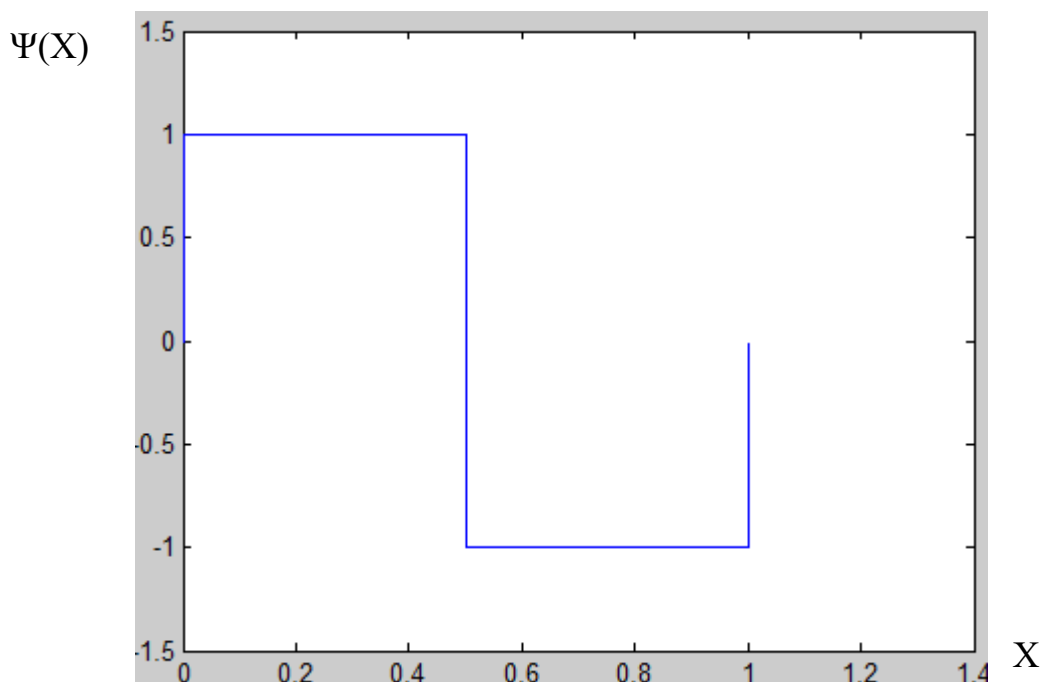


Рисунок 3 – Вейвлет Хаара

Вейвлеты Добеши (daubechie's wavelets) являются ортогональными вейвлетами с компактным носителем. Вейвлеты Добеши имеют разную размерность. В данной выпускной квалификационной работе будут использоваться вейвлеты Добеши2 и Добеши4.

Данные вейвлеты поддерживают дискретное преобразование. Недостатками являются недостаточная периодичность и

несимметричность[6]. Эти вейвлеты используются для устранения шумов на изображении.

На рисунке 4 представлен график вейвлета Добеши2, а на рисунке 5 – Добеши4.

$\Psi(X)$

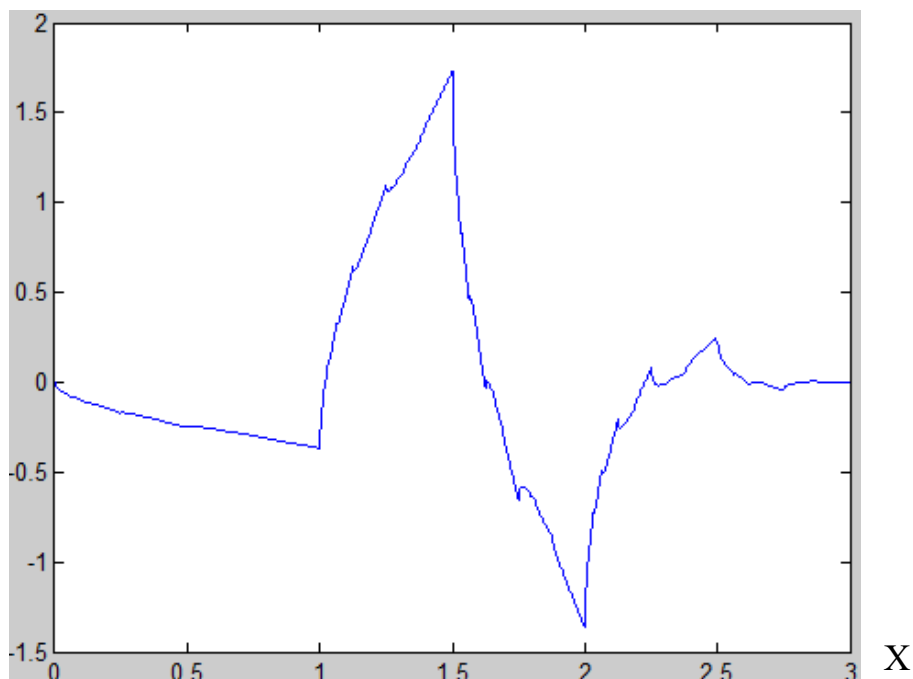


Рисунок 4 – График вейвлета Добеши2

$\Psi(X)$

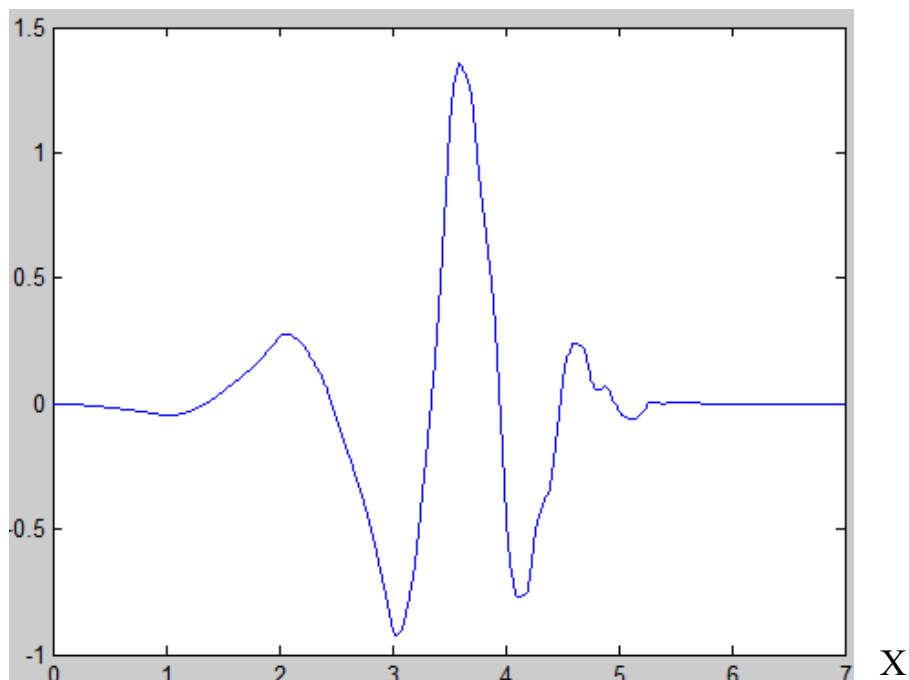


Рисунок 5 – График вейвлета Добеши4

Следующими используемыми вейвлетами являются симплеты. Симплет2 и симплет4 относятся к ортогональным вейвлетам и позволяют

восстанавливать сигнал, что требуется для возможности очистки сигнала или изображения. Симплеты близки к симметричности и являются недостаточно периодическими[6].

На рисунках 6 и 7 представлены графики симплета2 и симплета4 соответственно.

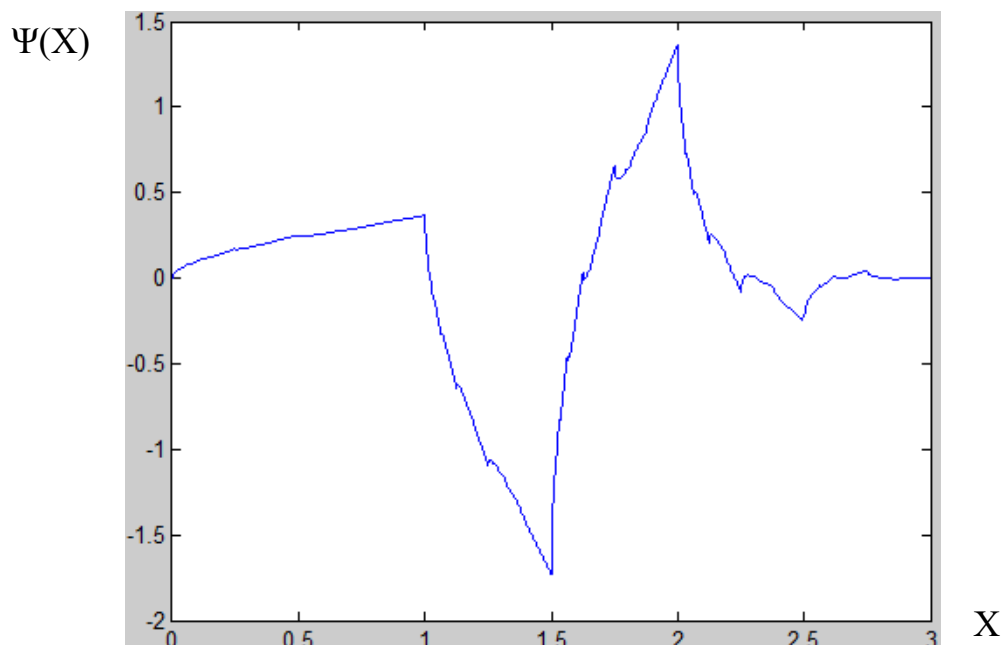


Рисунок 6 – График симплета2

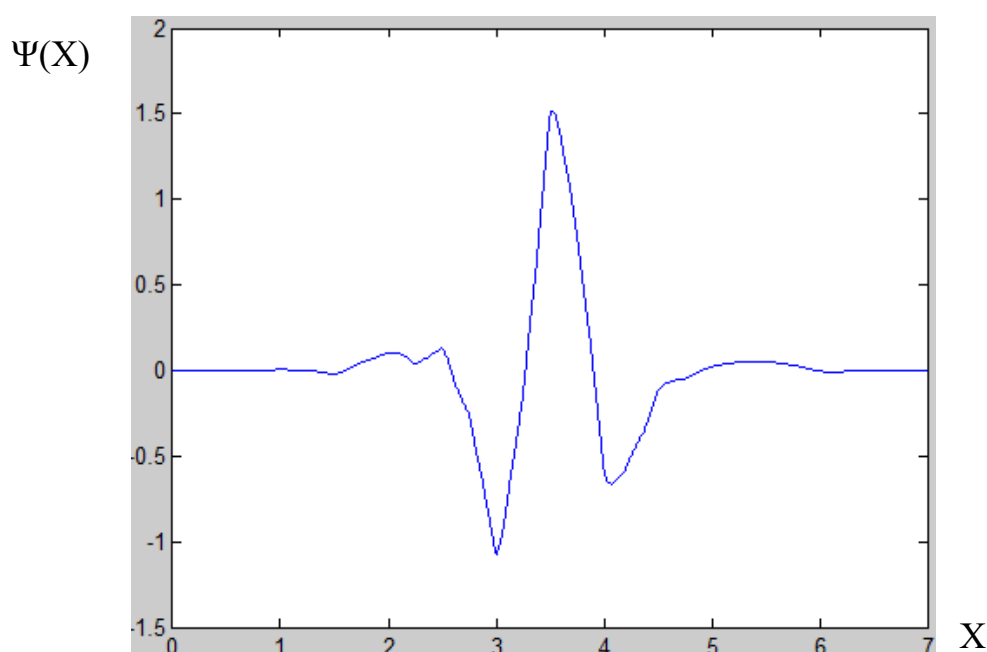
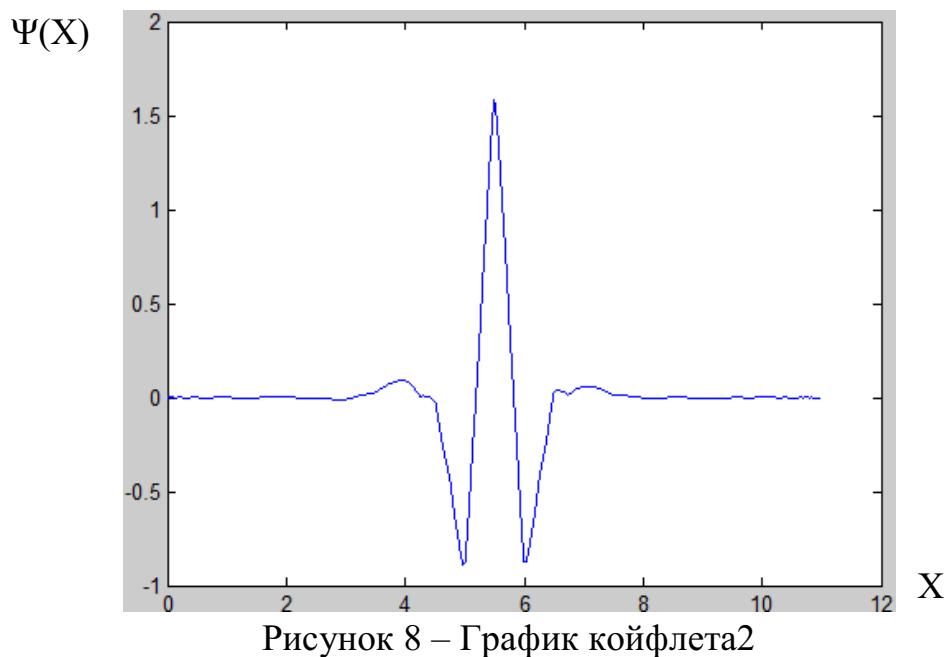
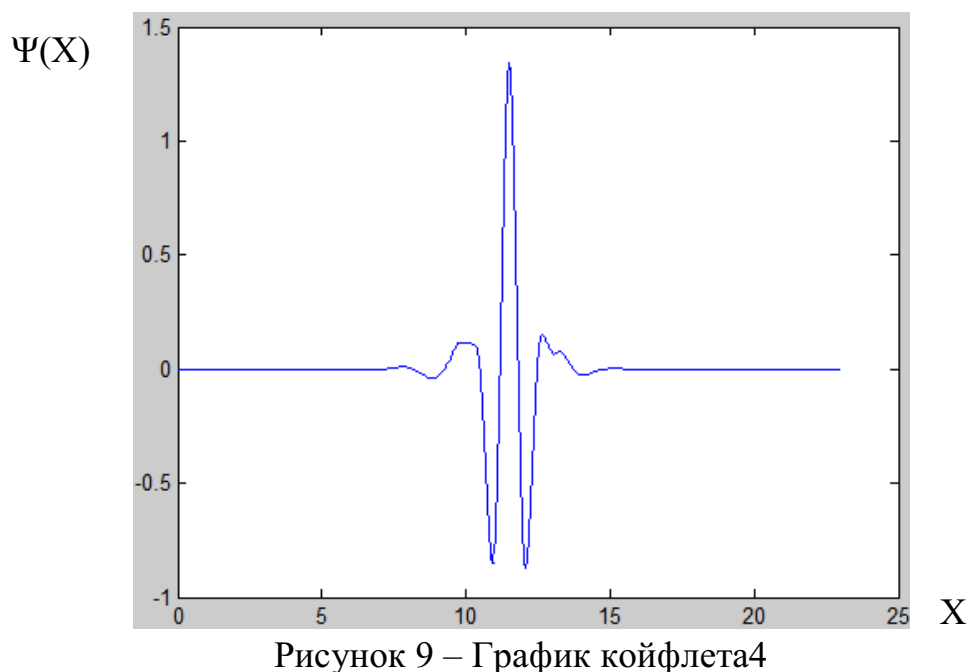


Рисунок 7 – График симплета4

Последней группой используемых вейвлетов являются койфлеты. Используемая в программе размерность – 2 и 4. Койфлеты, как и симплеты и вейвлеты Добеши, относятся к ортогональным вейвлетам с компактным

носителем. Они также являются ортогональными и поддерживают дискретное преобразование. Являются недостаточно периодическими и несимметричными. Важно отметить, что койфлеты могут работать с отрицательными коэффициентами.

На рисунках 8 и 9 представлены графики койфлета2 и койфлета4 соответственно.



Таким образом, были рассмотрены те вейвлеты, которые будут использоваться в разрабатываемой системе. Интерфейс будет предоставлять

выбор базового вейвлета. Таким образом, имеется возможность сопоставить полученные результаты.

1.3 Шумоподавление изображений

Разрабатываемая система должна устранять шум на изображении. Причины шумов могут быть различны, например, неисправность фотоаппаратуры или носителя, неподходящие погодные условия или потери при передаче изображения.

Выделяют следующие виды шума:

1. Соль и перец (случайные пиксели становятся черными или белыми).
2. Гауссов шум (колебания яркости пикселей, определяемые нормальным законом).
3. Мультипликативный шум (пиксели изображения умножаются на случайное значение).

В разрабатываемой программе необходимо реализовать описываемые методы фильтрации изображений. По полученным изображениям следует сопоставить результаты в зависимости от способа устранения шума и наложенного шума, и степени зашумленности.

Как было сказано ранее, wavelet-преобразование позволяет получить аппроксимирующие и детализирующие коэффициенты. Аппроксимирующая часть содержит существенную информацию об изображении, а детализирующая – детали изображения. Следовательно, шумовая составляющая имеется в детализирующих коэффициентах, значит, для устранения шума их можно отбросить[7].

Другим вариантом устранения шума и очистки изображения является трешолдинг (thresholding). Предполагается, что шумовая составляющая является значением, меньшим по модулю, чем основная информация. Это означает, что для очистки изображения можно сделать нулевыми коэффициенты, значение которых меньше определенного значения (порога). Данный метод называется пороговой обработкой или трешолдингом.

Пороговая обработка бывает двух видов: жесткая и мягкая. При жестком трешолдинге остаются неизменными коэффициенты, значения которых выше заданного порога или равны этому значению, а коэффициенты, значения которых меньше порога превращаются в нули. При мягком трешолдинге коэффициенты, значение которых меньше заданного порога, заменяются нулями, а остальные коэффициенты уменьшаются на величину порога.

Для фильтрации изображений применяется жесткая пороговая обработка. Мягкий трешолдинг в данном случае неприменим.

Для многоуровневого wavelet-преобразования используется многоуровневый трешолдинг. Возможно задание фиксированного значения порога для каждого уровня. Другим способом является задание различных порогов для различных уровней.

Результат фильтрации изображения, безусловно, зависит от значения порога шумоподавления. Если его значение невелико, то результат слабо отличим от исходного зашумленного изображения, так как превратятся в ноль не все шумовые составляющие. Если значение порога будет очень велико, то выходное изображение будет размытым, так как превратятся в ноль компоненты, несущие информацию.

Таким образом, алгоритм устранения шума на изображении будет состоять из трех следующих этапов.

1. На этапе декомпозиции выбирается базовый wavelet и уровень преобразования.
2. На этапе детализации в соответствии с заданным порогом выполняется трешолдинг детализирующих коэффициентов на каждом уровне преобразования.
3. На этапе реконструкции восстанавливается очищенное изображение по имеющимся аппроксимирующим коэффициентам и очищенным детализирующим коэффициентам.

Алгоритм очистки изображения от шума представлен на рисунке 10.

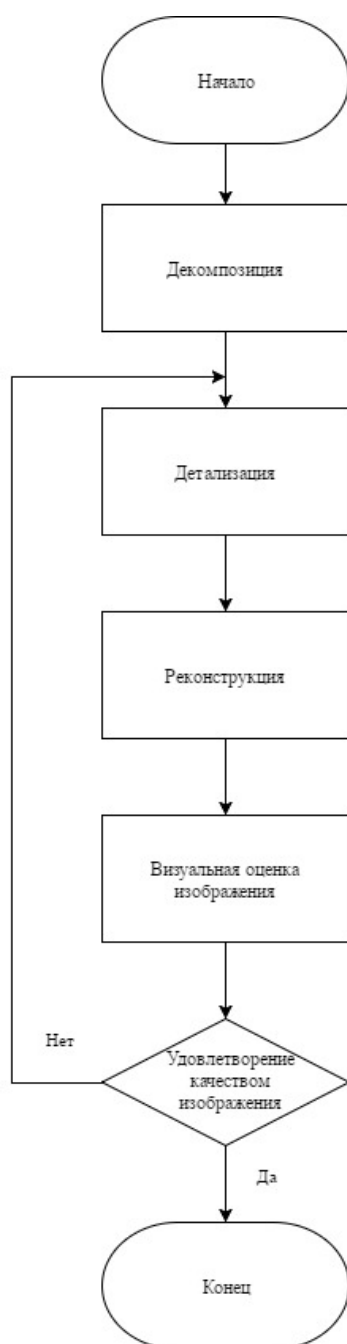


Рисунок 10 – Алгоритм очистки изображения от шума

Таким образом, для улучшения качества изображения, а именно, устранения шума при помощи wavelet-преобразования, необходимо выполнить следующие действия: выбрать исходное изображение, базовый wavelet, уровень wavelet-декомпозиции (преобразования), порог обработки. Программа должна выполнить действия, описанные ранее.

1.4 Обзор аналогичных методов

1.4.1 Преобразование Фурье

Преобразование Фурье является математическим способом решения рассматриваемой задачи.

Преобразование Фурье позволяет представить имеющийся сигнал тригонометрическими функциями. Данный метод основан на разложении функции в ряд. При выполнении преобразования необходимо рассчитать коэффициенты Фурье. Они представляют собой комплексные числа.

Достоинства преобразования Фурье:

1. При использовании больших коэффициентов сжатия получаются высокочастотные колебания, а при малых – низкочастотные[3].
2. Разложение в ряд Фурье допускает идеальную дискретизацию[3].
3. Преобразование Фурье позволяет перейти от временного представления сигнала к частотному[8].

Недостатки преобразования Фурье:

1. Преобразование требует знания сигнала не только в прошлом, но и в будущем[6]. Данное требование недостижимо, является абстракцией.
2. Точное восстановление сигнала после прямого и обратного преобразований Фурье теоретически и практически невозможно[6].
3. Численное интегрирование имеет вычислительные трудности.
4. Отдельные особенности сигнала (разрывы или пики) «размазываются» по частотной оси в результате преобразования[6]. Также использование синусоиды не позволяет представить резкие перепады сигналов или сигналы прямоугольной формы (импульсы).
5. «Плохо» работает на сигналах с размерностью более двух.

Вейвлеты хорошо локализованы во времени (или в пространстве)[6], что является преимуществом по сравнению с преобразованием Фурье. Они обеспечивают двумерную развертку исследуемого одномерного сигнала, появляется возможность анализировать сигнал в физическом (время,

координата) и в частотном пространствах[1]. Некоторые wavelet-преобразования, в отличие от преобразования Фурье, обеспечивают точное восстановление исходного сигнала, а также отражают особенности сигнала[6].

Важным критерием сопоставления аналогов является скорость вычислений. Стоит отметить следующие моменты.

Wavelet-преобразование привносит новые возможности в обработку сигналов или изображений, при этом скорость вычислений отходит на второй план не только по этой причине, но и вследствие высокой вычислительной мощности современных компьютеров.

В основе преобразования Фурье, как было сказано ранее, лежит синусоидальная функция. Ее вычисление труднее, чем вычисление многих вейвлетов, например, Хаара.

Аппарат преобразования Фурье использует комплексные числа, а wavelet-преобразование работает с дробными числами, следовательно, с этой точки зрения вычисление вейвлетов проще.

1.4.2 Линейный фильтр

Данный фильтр и следующие рассматриваемые фильтры не являются математическими способами решения рассматриваемой задачи.

Линейный фильтр включает в себя усредняющий фильтр и фильтр Гаусса.

В рассматриваемых фильтрах выделяют такое понятие, как апертура (или окно). Оно означает квадратную матрицу небольшого размера, внутри которой происходит обработка пикселей исходного изображения.

Работа линейного фильтра основана на усреднении значений внутри окна. Данный фильтр определяется вещественнозначной функцией, называемой ядром фильтра. Фильтрация осуществляется путем умножения ядра на элемент матрицы исходного изображения. В результате получается новое очищенное изображение.

При больших размерах окна усредняющий фильтр создает рамку по границам изображения, что недопустимо на современном этапе развития технологий. Помимо этого, линейный фильтр сглаживает перепады резкости. При решении требуемых задач это серьезный недостаток. Исключить его не представляется возможным, так как в изображениях во многих случаях будут иметься перепады яркости и переходы между текстурами. При небольших размерах окна линейный фильтр может дать неплохой результат, при этом, созданные границы будут малозаметны.

При устранении гауссовского шума линейный фильтр дает хороший результат [5], что является достоинством фильтра. Однако, применение рассматриваемого фильтра при других видах шума неэффективно [5].

Линейный фильтр предназначен для работы с линейными процессами. Следовательно, для современных задач он не позволит получить хороший результат, так как большинство рассматриваемых процессов в информатике носит нелинейный характер.

В целом, медианный и адаптивный фильтры дают лучший результат, чем линейный фильтр. Wavelet-преобразование является хорошей

					ВКР 090301.10. 17 81	Лист
Изм.	Лист	№ докум	Подпись	Дата		23

альтернативой линейному фильтру. Оно дает более качественное решение и обеспечивает более глубоким инструментарием при очистке изображения. Wavelet-преобразование не создает рамок на изображении и не размывает его. Вернее, размытие изображения при wavelet-преобразовании возможно в случае, если задан слишком высокий порог устранения шума. Но это ни в коем случае не является проблемой wavelet-преобразования. Это следствие используемого метода устранения шума (пороговая обработка – трешолдинг).

1.4.3 Медианный фильтр

Принцип работы нелинейных фильтров подобен линейным фильтрам. Нелинейные фильтры могут выполнять различные операции. Например, нелинейный фильтр может выполнять вычисление медианных значений элементов окрестности изображений[9].

Медианный фильтр выполняет нелинейную фильтрацию шумов на изображении. Медианный фильтр имеет окно (апертуру), о котором было сказано ранее. При помощи окна вычисляется медианное значение пикселей. В этом заключается работа медианного фильтра.

Медианный фильтр, в отличие от линейного, не создает рамок на изображении и не размывает его. Данный фильтр игнорирует экстремальные значения пикселей на изображении. Такой принцип может быть применен для ослабления шума на изображении.

Медианный фильтр представляет собой эвристический метод обработки. Данный алгоритм не является математическим решением строго сформулированной задачи [5] в отличие от wavelet-преобразования.

Медианный фильтр применяется для подавления мультипликативного шума. Он дает лучший результат по сравнению с линейным фильтром.

1.4.4 Адаптивный фильтр

Рассмотренные ранее фильтры не учитывают статистических особенностей изображения, таких как среднее значение и ее среднеквадратическое отклонение. Адаптивный фильтр Винера учитывает эти особенности в пределах окна (апертуры).

Адаптивный фильтр применяется для устранения гауссовского шума. Медианный фильтр показывает хороший результат при невысокой степени зашумленности[2]. Данный фильтр применим и для сильно зашумленных изображений.

В отличие от линейного фильтра, адаптивный фильтр сохраняет детали исходного изображения и не создает рамок.

Таким образом, при определенных условиях медианный и адаптивный фильтры могут показать достойные результаты. Важно отметить, что эти фильтры не являются математическим решением рассматриваемой задачи. По этой причине wavelet-преобразования представляются более применимым аппаратом для очистки изображения. Они устраняют шумы, не зависимо от их вида и степени зашумленности. Результат будет зависеть от заданных параметров wavelet-преобразования и трешолдинга.

1.4.5 Таблица сравнения аналогов

Результаты обзора аналогов сгруппированы и приведены в таблице 1. В таблице «+» означает хороший результат при очистке шума или хорошую скорость вычислений. «+/-» означает среднее качество результата очистки изображения или среднюю скорость. «-» означает плохой результат очистки изображений.

Таблица 1 – Сравнение аналогов.

Метод	Wavelet- преобразование	Преобразование Фурье	Линейный фильтр	Медианный фильтр	Адаптивн ый фильтр
Критерий					
Шум «соль и перец»	+	+/-	-	-	-
Гауссов шум	+	+/-	+	-	+
Мультипликативный шум	+	+/-	-	+	-
Скорость вычислений	+	+/-	+	+	+

1.5 Обоснование необходимости разработки

Графическое отображение информации используется повсеместно во всех сферах человеческой деятельности: в науке, искусстве, медицине, технике, образовании, военной сфере, строительстве, транспорте, в кибернетике и других различных сферах.

Современные устройства, например фотоаппараты, смартфоны, планшетные компьютеры, позволяют получать изображения: фотоснимки или вручную созданные графические материалы, обрабатывать и передавать их. По указанным ранее причинам могут возникать шумы на изображениях. Следовательно, вопрос устранения шумов и улучшение изображений в целом остается открытым.

Для задачи очистки изображения от шумов имеются определенные методы. Они были рассмотрены в предыдущих пунктах. Одним из этих методов является wavelet-преобразование. Оно было предложено впервые Хааром в начале прошлого столетия. Это научное направление было продолжено Добеши и Маллатом.

Не смотря на то, что wavelet-преобразования были предложены давно, в настоящее время в русскоязычной литературе имеется всего несколько десятков книг о вейвлетах. Они начали появляться сравнительно недавно. Количество статей и публикаций по этой теме превысило тысячу.

По словам отечественных авторов научных трудов о вейвлетах, в русскоязычном сегменте не хватает книг о вейвлетах, а именно об их практическом применении.

В иностранной литературе высказан тезис о том, что сами wavelet-преобразования, их возможности, свойства и практическое применение до конца не рассмотрены.

Wavelet-преобразование является актуальной научной темой. Она имеет математическую основу и широкое практическое применение.

В данной выпускной квалификационной работе требуется рассмотреть wavelet-преобразования и их применение для очистки изображения от шумов.

Интерес также представляет сравнение результатов устранения шумов с использованием wavelet-преобразования с уже имеющимися способами. Некоторые из них были рассмотрены в предыдущих пунктах.

Важным моментом является полная оценка результатов и их зависимости от входного изображения (вида шума на изображении и степени зашумленности), от выбранного базового вейвлета, уровня wavelet-декомпозиции и порога трешолдинга.

Для демонстрации и анализа рассматриваемых возможностей wavelet-преобразования необходимо разработать программу, реализующую указанный математический аппарат. Возможности аппарата и требования к нему будут рассмотрены далее.

1.6 Формализованная постановка задачи

1.6.1 Требования к функциональным характеристикам

Функциональные характеристики определяют набор действий, выполняемых программой.

Как было сказано ранее, разрабатываемая система предназначена для повышения качества изображений за счет wavelet-преобразований.

Система должна выполнять следующие действия:

1. Загрузка изображения.
2. Выбор базового вейвлета.
3. Определение уровня wavelet-декомпозиции.
4. Визуализация результатов.
5. Сохранение очищенного изображения.

Для возможности сопоставления аппарата wavelet-анализа с имеющимися методами очистки изображений от шума, следует разработать компоненты, отвечающие за линейную фильтрацию (усредняющий фильтр и фильтр Гаусса), медианную фильтрацию и адаптивную фильтрацию (фильтр Винера).

Для более точной оценки необходимо разработать компонент, позволяющий накладывать различные шумы на изображение в различной степени.

1.6.2 Требования к надежности

Надежность характеризует способность программного обеспечения работать без отказов с заданной степенью достоверности.

Разрабатываемая программа, безусловно, должна выполнять свои функциональные возможности в полной мере.

Но, тем не менее, следует предусмотреть возможные ошибки:

1. Невозможно запустить диалог открытия файла.

Данная ситуация может быть вызвана какими-то системными ошибками.

2. Невозможно открыть изображение.

Файл, содержащий изображение, может быть занят другим процессом, то есть изображение может быть открыто на изменение в другой программе.

3. Невозможно запустить диалог сохранения файла.

Ситуация аналогична первому пункту.

4. Невозможно сохранить файл.

Причиной также являются системные ошибки, некорректный путь или недостаточность прав пользователя.

5. Введены некорректные параметры wavelet-преобразования.

Следует предусмотреть ввод только корректных параметров или отслеживать входные параметры в программе.

Таким образом, программа должна удовлетворять заданным функциональным характеристикам. Корректность работы приложения не должна зависеть от входных параметров. Программа должна отслеживать или запрещать ввод некорректных параметров.

1.6.3 Требования к составу и параметрам технических средств

Определим минимальные требования к аппаратной составляющей компьютера для работы с разработанным приложением:

- 32-разрядный (x86) или 64-разрядный (x64) процессор с тактовой частотой 1 ГГц или выше[11].
- 1 ГБ (для 32-разрядного процессора) или 2 ГБ (для 64-разрядного процессора) ОЗУ[11].
- Графическое устройство DirectX 9 с драйвером WDDM 1.0 или более поздней версии.
- Так же потребуется 5.1 ГБ для установки математического пакета.
- 50 МБ свободной памяти для .NET Framework.

О программных компонентах будет сказано в следующем пункте.

1.6.4 Требования к информационной и программной совместимости

Программа разрабатывалась и тестировалась на операционной системе «Windows 8». Программа будет поддерживаться следующими версиями операционной системы «Windows»: 7, 8, 8.1, 10.

Для работы с разработанным приложением потребуется наличие системы компьютерной математики Matlab 2011 года. Для ее установки требуется наличие 5.1 ГБ свободного дискового пространства.

Так же для работы приложения потребуется наличие установленной платформы .NET Framework 4.5. Она займет 50 МБ дискового пространства.

2 ОБОСНОВАНИЕ СРЕДСТВ РАЗРАБОТКИ

2.1 Выбор операционной системы

Операционная система предоставляет среду выполнения программы. Следовательно, для разработки программы, в первую очередь, необходимо выбрать операционную систему.

Данное программное обеспечение разрабатывалось и тестировалось на операционной системе «Windows 8». Как было отмечено ранее, описываемая программа будет работать и на других версиях названной операционной системы: 7, 8, 8.1, 10. Выбор операционной системы обусловлен следующими причинами.

По статистике ирландского веб-сервиса «StatCounter» на апрель 2017 года, процент пользователей операционной системы «Windows» для настольных персональных компьютеров составляет примерно 84%[12], что является наивысшим результатом. В данной статистике присутствовали операционные системы Mac OS X, Linux и Chrome OS.

Учитывая другие операционные системы (например, мобильные), популярность «Windows» составляет 42%[12].

Рассматриваемый сервис представляет собой инструмент анализа интернет-трафика (например, с каких операционных систем или браузеров пользователи посещают сайт). Статистика собирается с более трех миллионов сайтов.

Популярность операционной системы «Windows» в России обусловлена еще и тем, что поставщики компьютерной техники комплектуют ноутбуки и настольные персональные компьютеры установленной лицензионной версией рассматриваемой операционной системы. Помимо этого, «Microsoft» предоставлял возможность бесплатно установить операционную систему «Windows 10» до лета 2016 года. Несомненно, такой маркетинг повлиял на статистические показатели.

Платформа «Windows» поддерживает создание приложений на языках высокого уровня (C, C++, C# и др.). Следовательно, данная платформа поддерживает и интегрированные среды разработки. Для данной задачи требуется использовать язык программирования высокого уровня. Также необходима среда разработки, предоставляющая удобные инструментальные возможности для программиста. Это позволит разработать систему в соответствии с требованиями, отладить ее и минимизировать вероятность сбоя. О выборе языка программирования и среды разработки будет сказано далее.

Помимо этого, «Windows» поддерживает различные инструментальные программные средства, например, системы компьютерной математики. Это необходимо для создания программной системы в рамках данной выпускной квалификационной работы.

2.2 Выбор системы компьютерной математики

Система компьютерной математики представляет собой набор программных инструментальных средств различного назначения. В целом, такие системы выполняют самые разнообразные узкоспециализированные математические вычисления. Системы компьютерной математики, как правило, имеют свой язык программирования, который позволяет писать команды, вызывать функции, проводить расчеты и вычисления.

В рамках данной выпускной квалификационной работы требуется проводить сложные математические вычисления wavelet-преобразований изображений. Также в данной работе рассматриваются фильтры: линейный, медианный и адаптивный. Ручная реализация всех этих методов длительна и трудоемка. Целесообразно проводить математические вычисления в системе компьютерной математики. В соответствии с поставленной задачей, следует выбрать СКМ «Matlab» версии 2011a.

Данная система компьютерной математики ориентирована на матричные операции, что удобно в данном случае. Изображение в «Matlab» представляет собой матрицу. Черно-белое изображение представляется двумерной матрицей, а цветное – трехмерной. Wavelet-преобразования будут описываться матричными процедурами. Следовательно, ориентация «Matlab» на матричные операции привлекательна в данном случае.

«Matlab» создан таким образом, что любые вычисления производятся в режиме прямых вычислений, т.е. без подготовки[6]. Это делает Matlab мощным математическим пакетом. Такая реализация удобна для разработки рассматриваемой системы в рамках данной выпускной квалификационной работы.

Таким образом, «Matlab» работает, как интерпретируемый язык программирования. Его действия выполняются последовательно в матричном виде.

Помимо этого, для создания программ, «Matlab» предоставляет внешние интерфейсы. Это удобно для разработки приложений, т.к. появляется возможность использовать сложные математические вычисления.

Используемый язык программирования должен поддерживать внешние интерфейсы. Это влияет на выбор языка. Выбранный язык программирования и среда разработки будут описаны в следующем пункте.

2.3 Выбор языка программирования и среды разработки

В соответствии с выбранной операционной системой и системой компьютерной математики, был выбран язык программирования C#.

C# представляет собой гибрид из нескольких языков, он является таким же синтаксически чистым, как и Java, почти столь же простым, как VB, и практически таким же мощным и гибким, как C++ [10].

Рассматриваемая задача в рамках выпускной квалификационной работы носит прикладной характер, следовательно, необходимо использовать язык высокого уровня. Таким языком является C#.

Данный язык поддерживается платформой Windows. Он позволяет создавать приложения с удобным интерфейсом пользователя.

Язык программирования C# поддерживает внешние интерфейсы. В частности, имеется возможность подключить «Matlab», как внешний СОМ-объект. Следует отметить, что C# позволяет работать таким образом не только с системой компьютерной математики «Matlab», но и с другими программами, например «Word», «Excel» или «1С».

Язык программирования C# поддерживает объектно-ориентированную методологию разработки. Это позволяет использовать основные парадигмы объектно-ориентированного программирования: инкапсуляцию, наследование и полиморфизм. Такой подход предоставляет возможность восходящей разработки программы, т.е. сначала разработать отдельные компоненты (классы), затем – компоненты, использующие созданные классы. Выбранная методология позволяет спроектировать разрабатываемую систему в терминах объектно-ориентированного программирования.

Используя данный метод, можно упростить и оптимизировать процесс разработки и уменьшить сложность, разделив задачу на отдельные подзадачи. Также это позволит упростить внесение изменений в программу и, при необходимости, сопровождение программного обеспечения.

В соответствии с выбранной операционной системой «Windows» и языком программирования С# необходимо использовать среду разработки Microsoft Visual Studio. Используемая версия – 2012.

Данная среда поддерживает различные языки программирования, в частности, язык высокого уровня С#. Также среда предоставляет широкие возможности для разработки: кодирование, трассировка, отладка и тестирование. Среда обеспечивает возможность модульной разработки, что удобно и необходимо для создания сложных объектно-ориентированных приложений.

Таким образом, для создания системы wavelet-преобразования изображений, был выбран язык программирования С# и среда разработки Microsoft Visual Studio 2012.

3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Входные и выходные данные

Входными данными системы является изображение. Допустимые форматы *.jpg, *.bmp, *.png.

В программе предусмотрена возможность принудительного зашумления изображения. Программа позволяет ввести параметры wavelet-преобразования и трешолдинга, ввести параметры для фильтров, очистить изображение и сохранить изображение.

В процессе выполнения программы исходное изображение представляется трехмерным массивом. Он содержит значения каждой цветовой составляющей (красной, синей, зеленой). При wavelet-преобразовании данный массив раскладывается на аппроксимирующие и детализирующие коэффициенты. Их количество зависит от уровня декомпозиции.

Выбранный wavelet, уровень преобразования, значение порога обработки являются входными данными. Они задаются в программе при помощи соответствующих элементов управления. Корректность этих параметров контролируется этими элементами управления и их обработчиками.

При использовании фильтров изображение обрабатывается апертурой (окном). Размерность окна является также исходным значением и задается в программе при помощи пользовательского интерфейса.

Результатом работы программы является очищенное изображение. Формат изображения: *.jpg, *.bmp или *.png. Программа позволяет сохранять очищенные изображения в указанных форматах.

Таким образом, входными данными для программы являются: изображение, его путь, параметры wavelet-декомпозиции, трешолдинга и размеры окна для линейного, медианного и адаптивного фильтров, а так же путь для очищенного изображения.

3.2 Разработка класса wavelet-преобразования изображения

Wavelet-преобразование выполняется средствами системы компьютерной математики «Matlab». Для этого необходимо, во-первых, разработать и отладить код в среде «Matlab». Во-вторых, следует подключить «Matlab», как внешний СОМ-объект в создаваемую программу. В-третьих, необходимо разработать компонент на языке С#, который будет выполнять полученный на первом шаге код в среде «Matlab».

В соответствии с изложенной ранее теорией wavelet-преобразований, был разработан код в системе компьютерной математики «Matlab» для реализации wavelet-преобразования изображений. Разрабатываемая система позволяет вручную задавать параметры wavelet-декомпозиции (wavelet, уровень преобразования, коэффициент пороговой обработки). В первом случае коэффициент пороговой обработки одинаков для всех уровней декомпозиции. Имеется второй вариант работы wavelet-преобразования, в котором коэффициент пороговой обработки определяется программой автоматически для каждого уровня. Для двух случаев имеется разный код соответственно.

В первом случае пример разработанного кода при использовании вейвлета Хаара, уровня преобразования, равного 5, и коэффициента пороговой обработки, равного 52, в среде «Matlab» выглядит следующим образом:

```
>> X = imread('C:\Users\Alexander\Desktop\image.jpg');  
>> w = 'haar';  
>> n = 5;  
>> thr = 52;  
>> [c, l] = wavedec2(X, n, w);  
>> xd = wdencomp('gbl', c, l, w, n, thr, 'h', 1);  
>> xd = round(xd);  
>> if isequal(class(X), 'uint8'), xd = uint8(xd); end  
>> subplot(221), image(X), title('Original image')
```

```
>> subplot(222), image(xd), title('De-noised image')
>> imwrite(xd, 'C:\Users\Alexander\Desktop\clear image.jpg');
```

Представленный код выполняет следующие действия: чтение изображения «image.jpg», его wavelet-преобразования до заданного уровня с заданным вейвлетом, жесткую пороговую обработку с заданным коэффициентом трешолдинга, вывод изображений на экран и сохранение очищенного изображения с именем «clear image.jpg».

Функция `imread` читает изображение, путь к которому задан символической константой в одинарных кавычках. Функция может работать с полутоновыми или полноцветными изображениями. В данном случае на вход подаются полноцветные изображения. В результате этого, в переменной `X` хранится трехмерный массив. Его элементы описывают три цветовые составляющие (красную, зеленую, синюю).

Следующие три строки задают имя вейвлета, в данном случае вейвлета Хаара (Haar wavelet), уровень преобразования `n` и коэффициент пороговой обработки `thr`.

Функция `wavedec2` отвечает за wavelet-преобразование уровня `n` исходного изображения `X` с использованием вейвлета `w`. Функция возвращает вектор декомпозиции `C` и соответствующую учетную матрицу `L` [6].

Функция `wdencmp` возвращает очищенный массив `xd` (изображение) после пороговой обработки. Параметр `'gbl'` задает фиксированное значение порога трешолдинга для каждого уровня. `s` и `l` – результаты wavelet-декомпозиции изображения на предыдущем шаге. `w` – имя вейвлета, `n` – уровень wavelet-преобразования, `thr` – значение коэффициента пороговой обработки, `'h'` – жесткая пороговая обработка, `1` – коэффициенты аппроксимации не могут ограничиваться.

Далее полученная очищенная матрица округляется и осуществляется приведение к типу `uint8`.

Следующие две строки выводят исходное и очищенное изображения соответственно.

Функция `imwrite` сохраняет массив в виде полноцветного изображения. Полученное изображение можно открывать в любом графическом редакторе.

Теперь следует рассмотреть второй случай, в котором значение коэффициента пороговой обработки определяется программой автоматически для каждого уровня декомпозиции.

Пример кода представлен ниже. Как и в предыдущем примере, используется вейвлет Хаара, уровень преобразования 5. Коэффициент пороговой обработки определяется автоматически для каждого уровня.

```
>> X = imread('C:\Users\Alexander\Desktop\image.jpg');
>> w = 'haar';
>> n = 5;
>> [c, s] = wavedec2(X, n, w);
>> alpha = 3;
>> m = 2.7*prod(s(1,:));
>> thr = wdcbm2(c, s, alpha, m);
>> xd = wdencmp('lvd', X, w, n, thr, 'h');
>> xd = round(xd);
>> if isequal(class(X), 'uint8'), xd = uint8(xd); end
>> subplot(221), image(X), title('Original image')
>> subplot(222), image(xd), title('De-noised image')
>> imwrite(xd, 'C:\Users\Alexander\Desktop\clear image.jpg');
```

Функция `imread` читает входное изображение и представляет его в виде трехмерного массива в переменной `X`.

Далее задаются `wavelet` и уровень `wavelet`-декомпозиции.

Функция `wavedec2` выполняет `wavelet`-преобразование с заданными параметрами.

Далее задаются параметры для функции `wdcbm2`. Значение $\alpha = 3$ типично для очистки изображения[6]. Значение m находится в рекомендуемом диапазоне от $\text{prod}(s(1,:))$ до $6 \cdot \text{prod}(s(1,:))$ [6].

Функция `wdcbm2` возвращает вектор коэффициентов пороговой обработки `thr` для удаления шумов. Входными параметрами также являются результаты `wavelet`-декомпозиции `c` и `s`.

Функция `wdencmp` возвращает очищенный массив (изображение). Параметр `'lvd'` означает, что для каждого уровня используется разный порог. Остальные параметры аналогичны предыдущему случаю. Стоит отметить, что здесь также осуществляется жесткая пороговая обработка.

В следующих строках массив округляется и приводится к типу `uint8`. Далее выводятся исходное изображение и очищенное. Затем сохраняется очищенное изображение.

Таким образом, был разработан и отлажен код в системе компьютерной математики «Matlab». Код выполняет загрузку изображения, ввод параметров, `wavelet`-очистку и сохранение изображения. Входные параметры будут задаваться в разрабатываемой программе. Для примера кода они заданы вручную.

Следующим шагом необходимо подключить в проект Matlab, как СОМ-объект. Для этого необходимо нажать Проект->Добавить ссылку и добавить Matlab, как СОМ-объект. Это отображено на рисунке 11.

Это даст возможность использовать в проекте класс `MAppClass`. Этот класс предназначен для работы с «Matlab». Особый интерес представляют три метода этого класса.

`MAppClass.Execute(string Name)` выполняет команду `Name` на языке Matlab. Команда задается в двойных кавычках, как строка.

`MAppClass.GetVariable(string Name, string Workspace)` позволяет получить переменную `Name` из рабочего поля `Workspace`.

MyMatlab.PutWorkspaceData(string Name, string Workspace, object Data)
 позволяет поместить переменную object с именем Name в окружение Matlab.

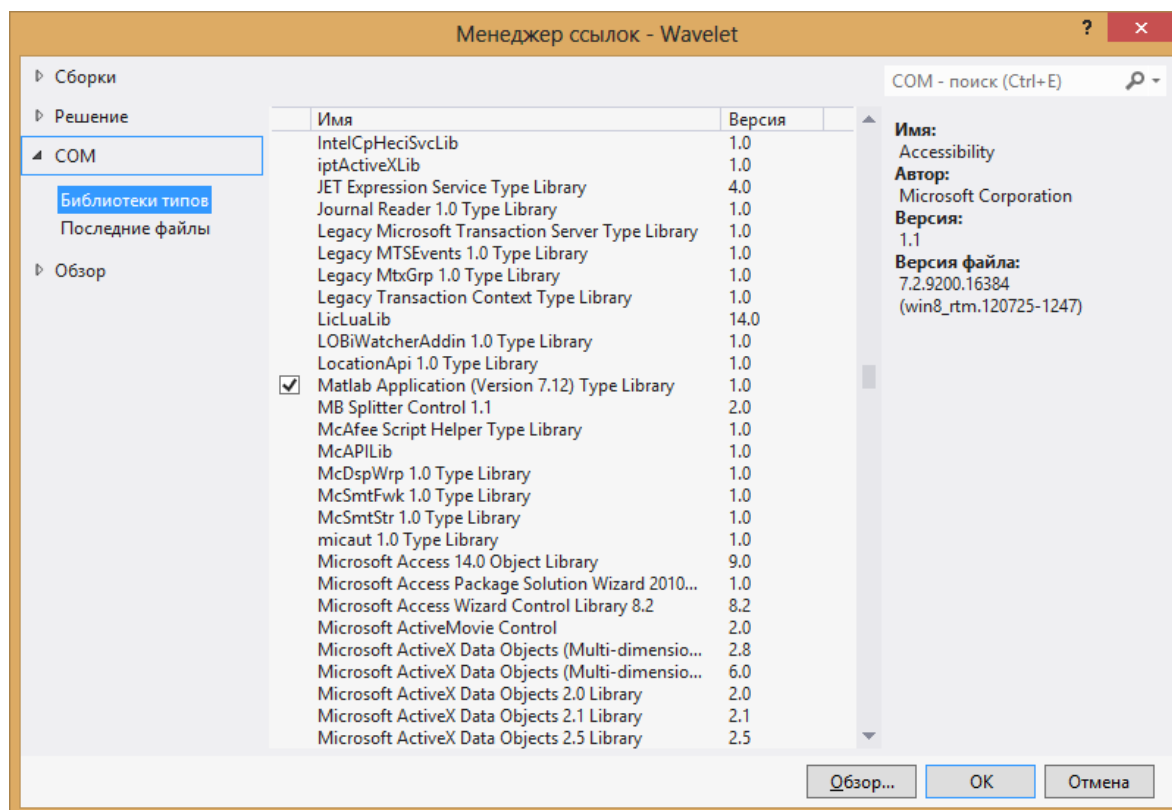


Рисунок 11 – Подключение Matlab'а в проект

Следующим шагом необходимо разработать компонент, который будет выполнять код на языке Matlab, для создаваемой программы с учетом специфики задачи.

Для этой цели следует разработать класс. В данном классе будет объявлен атрибут – объект класса MLAppClass. Также следует предусмотреть атрибут, который будет показывать, было ли очищено изображение. Таким образом, начинается разработка класса wavelet-преобразования и трешолдинга изображения.

Класс будет содержать пять методов: очистка рабочего пространства Matlab, загрузка изображения, фильтрация изображения с заданным порогом трешолдинга, фильтрация изображения с автоматическим определением порога трешолдинга и сохранение очищенного изображения.

Очистка рабочего пространства Matlab заключается в удалении объявленных и измененных ранее переменных. Это необходимо для предотвращения возможных ошибок. Процедура выполняется командой «clear all» в среде Matlab. В разработанном классе соответствующий метод называется ClearWorkSpace().

Метод загрузки изображения получает на вход адрес изображения. Этот адрес используется для открытия изображения в рабочем пространстве Matlab с помощью функции imread, которая рассматривалась ранее. Метод называется OpenImage(string ImageWay).

Метод фильтрации изображения с заданным порогом трешолдинга получает на вход имя вейвлета, уровень преобразования и порог трешолдинга. Метод выполняет фильтрацию изображения. Разработанный метод получил название DeNoising(string WaveName, int Level, int Threshold).

Метод фильтрации изображения с автоматическим определением порога трешолдинга получает на вход имя вейвлета и уровень декомпозиции. Метод выполняет аналогичную фильтрацию изображения за исключением того, что выбор коэффициента пороговой обработки осуществляется автоматически для каждого уровня. Описанная процедура выполняется методом DeNoisingAutoThrsh(string Wavelet, int Level).

Метод сохранения изображения получает на вход путь к изображению. Этот путь использует функция imwrite, которая сохраняет очищенный массив в виде изображения. Для выполнения данной процедуры следует вызвать метод SaveClear(string SaveFile).

Таким образом, был разработан класс «WaveletClass.cs», который выполняет фильтрацию изображения, используя wavelet-преобразование. Декомпозиция выполняется при помощи системы компьютерной математики «Matlab». Листинг класса представлен в приложении А.

3.3 Разработка компонентов наложения шума

Для удобства оценки результатов работы программы, целесообразно разработать компоненты, отвечающие за наложение шума на изображение.

В этих целях следует разработать отдельную форму, содержащую элементы управления, и отдельный класс, выполняющий наложение шума.

Как было отмечено ранее, существует три основных вида шума: соль и перец, Гауссов шум и мультипликативный шум. В программе будет предусмотрена возможность выбора шума и выбора степени зашумленности. Степень зашумленности будет задаваться статически, так как входными параметрами для данных методов будут математические термины, а интерфейс должен скрывать детали реализации от пользователя.

Внешний вид созданного окна представлен на рисунке 12.

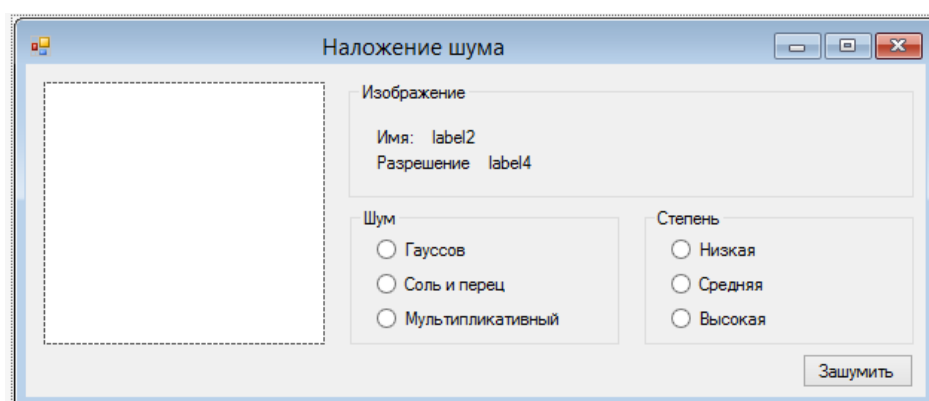


Рисунок 12 – Конструктор формы наложения шума на изображение

Форма будет выводить исходное изображение слева для удобства пользователя. Будет предоставлен выбор одного шума и одной степени зашумленности. Сверху формы выводятся имя изображения и разрешение.

За выполнение названных действий будет отвечать класс «Noise.cs». Его листинг представлен в приложении Б. Листинг рассмотренной формы «NoiseForm.cs» изложен в приложении В.

В классе наложения шума содержится три метода для трех шумов соответственно. В методах определяется степень зашумленности и происходит непосредственно наложение шума. Как и в предыдущем пункте, подобные действия выполняются при помощи системы компьютерной математики «Matlab».

Ниже представлен пример кода в среде Matlab для наложения шума вида соль и перец в средней степени.

```
>> N = imread('C:\Users\Alexander\Desktop\tiger.jpg');  
>> I = imnoise(N, 'salt & pepper', 0.07);  
>> imshow(I)  
>> imwrite(I, 'C:\Users\Alexander\Desktop\image.jpg');
```

Основное действие выполняет функция `imnoise`. Она принимает на вход исходное изображение `N`, символьную константу `'salt & pepper'`, обозначающую вид шума, а также плотность шума. В данном случае она равна 0.07. Именно эта деталь скрывается от пользователя.

Для наложения Гауссова шума в средней степени пример кода будет выглядеть аналогично.

```
>> N = imread('C:\Users\Alexander\Desktop\tiger.jpg');  
>> I = imnoise(N, 'gaussian', 0.1, 0.04);  
>> imshow(I)  
>> imwrite(I, 'C:\Users\Alexander\Desktop\image.jpg');
```

Функция `imnoise` в данном случае принимает на вход не только исходное изображение для зашумления и вид шума, но и математическое ожидание `m` и дисперсию `v`. Эти детали заданы статические для трех случаев и скрываются от пользователя.

Для создания мультипликативного шума средней степени пример кода приведен ниже.

```
>> N = imread('C:\Users\Alexander\Desktop\tiger.jpg');  
>> I = imnoise(N, 'speckle', 0.07);  
>> imshow(I)  
>> imwrite(I, 'C:\Users\Alexander\Desktop\image.jpg');
```

В данном примере функция `imnoise` принимает исходное изображение и вид шума, а так же дисперсию `v`.

Таким образом, разработан класс, выполняющий описанные действия на языке C#, используя при этом код в среде Matlab.

3.4 Разработка фильтров

Для разработки линейного, медианного и адаптивного фильтров используется механизм открытого наследования. Данные фильтры наследуются от абстрактного класса фильтра. Такой подход предоставляет возможность наследования интерфейса базового класса производными классами. Следовательно, в программе становится возможным повторное использование кода.

Открытое наследование описывает отношение is-a между классами. Это означает, что производный класс является базовым классом в том числе в терминах анализируемой предметной области, т.е. линейный, медианный и адаптивный фильтры являются фильтрами.

Для названных трех фильтров создан абстрактный класс «Filter.cs». Он содержит конструктор и два метода.

Конструктор инициализирует логическую переменную isClear ложным значением. Переменная означает, было ли очищено изображение или нет.

Первый метод выставляет размер окна (апертуры), второй метод выполняет сохранение очищенного изображения.

Все перечисленные действия являются общими, следовательно, целесообразно, используя возможности объектно-ориентированного программирования, сократить объем кода и исключить написание одинаковых действий для каждого фильтра. Листинг абстрактного класса «Filter.cs» представлен в приложении Г.

Все три фильтра выполняются в одной и той же форме (окне). Оно запускается с входными параметрами – тип фильтра, имя входного изображения и его разрешение. Это один из механизмов передачи данных между формами.

Линейный фильтр представлен усредняющим фильтром и фильтром Гаусса. Их отличия, достоинства и недостатки были описаны в первом разделе.

За работу фильтров, как и в предыдущих случаях, отвечает система компьютерной математики «Matlab».

Пример кода усредняющего фильтра для окна 3x3 приведен ниже.

```
>> X = imread('C:\Users\Alexander\Desktop\image.jpg');  
>> h = fspecial('average', [3 3]);  
>> i = imfilter(X, h);  
>> imshow(i)  
>> imwrite(i, 'C:\Users\Alexander\Desktop\clear image.jpg');
```

В данном примере стоит обратить внимание на две функции `fspecial` и `imfilter`. Функция `fspecial` возвращает маску `h` двумерного фильтра[6]. Маска предназначена для хранения коэффициентов фильтра. Маска передается в функцию `imfilter`, которая фильтрует изображение `X` в соответствии с маской `h`.

Для фильтра Гаусса, который также относится к линейному, пример кода приведен ниже.

```
>> X = imread('C:\Users\Alexander\Desktop\image.jpg');  
>> h = fspecial('gaussian', [3 3]);  
>> i = imfilter(X, h);  
>> imshow(i)  
>> imwrite(i, 'C:\Users\Alexander\Desktop\clear image.jpg');
```

Функция `fspecial` принимает два аргумента. Первый – тип фильтра, второй – размер окна (апертуры). Функция возвращает, как было сказано ранее, маску двумерного фильтра. Функция `imfilter` очищает изображение `X` в соответствии с маской `h`.

Разработанный код класса линейного фильтра «LinearFilter.cs» находится в приложении Д.

Для медианного фильтра был разработан отдельный класс «MedianFilter.cs». Листинг класса находится в приложении Е.

Пример кода для медианного фильтра приведен далее.

```

>> X = imread('C:\Users\Alexander\Desktop\image.jpg');
>> R = X(:,:,1);
>> G = X(:,:,2);
>> B = X(:,:,3);
>> i(:,:,1) = medfilt2(R, [3 3]);
>> i(:,:,2) = medfilt2(G, [3 3]);
>> i(:,:,3) = medfilt2(B, [3 3]);
>> imshow(i)
>> imwrite(i, 'C:\Users\Alexander\Desktop\clear image.jpg');

```

В данном случае исходное изображение делится на три цветовые составляющие: красную, синюю, зеленую. Эти составляющие очищаются медианным фильтром (функция `medfilt2`). Входными параметрами функции являются исходное изображение или его часть и размер окна.

Для адаптивного фильтра Винера был разработан свой класс «AdaptiveFilter.cs». Листинг класса находится в приложении Ж. Работа класса основывается на следующем примере.

```

>> X = imread('C:\Users\Alexander\Desktop\image.jpg');
>> R = X(:,:,1);
>> G = X(:,:,2);
>> B = X(:,:,3);
>> i(:,:,1) = wiener2(R, [3 3]);
>> i(:,:,2) = wiener2(G, [3 3]);
>> i(:,:,3) = wiener2(B, [3 3]);
>> imshow(i)
>> imwrite(i, 'C:\Users\Alexander\Desktop\clear image.jpg');

```

Как и в предыдущем случае, работа класса основывается на разделении изображения на три цветовые составляющие. Эти цветовые составляющие являются первыми аргументами функции `wiener2` (адаптивного фильтра Винера). Вторым аргументом является размер окна.

Таким образом, был разработан код в среде Matlab, на основе которого были созданы классы для каждого фильтра. Каждый класс содержит метод очистки изображения и метод сохранения изображения. Конструктор и общие действия наследуются от абстрактного класса «Filter.cs».

3.5 Разработка главного окна

Завершающим этапом разработки является создание главного окна и реализация его действий. Ранее были созданы классы wavelet-преобразований и классы линейного, медианного и адаптивного фильтров. Теперь необходимо реализовать действия, выполняемые при помощи этих классов.

Для начала необходимо разместить и скомпоновать элементы управления. Следует учесть необходимость удобства интерфейса и его понятность для пользователя.

Конструктор окна изображен на рисунке 13.

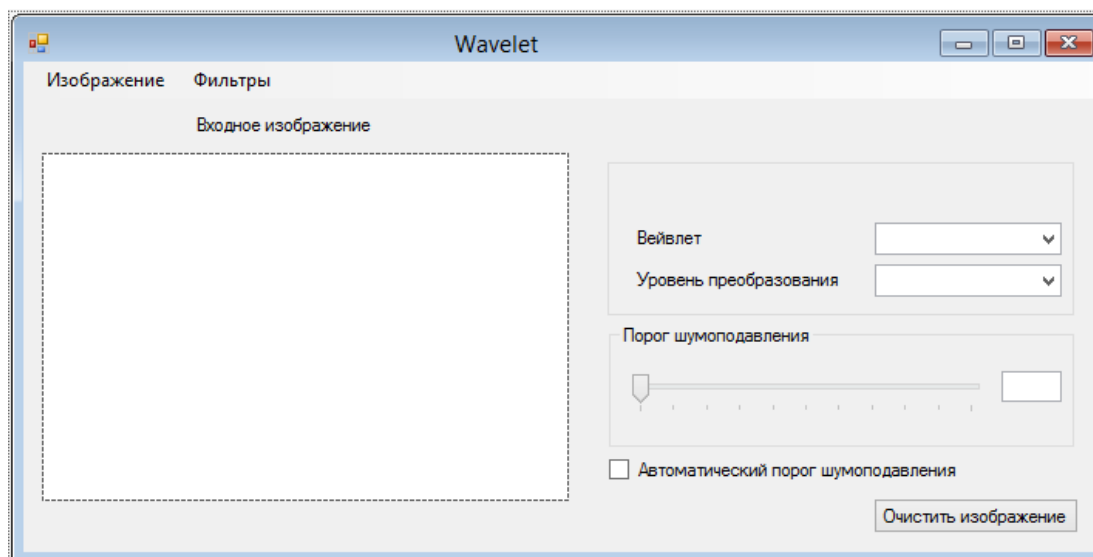


Рисунок 13 – Конструктор главного окна

В разделе «Изображение» находятся четыре вкладки: «Открыть», «Наложить шум», «Сохранить очищенное» и «Закрыть».

В разделе «Фильтры» помещено четыре вкладки: «Усредняющий», «Гаусса», «Медианный» и «Адаптивный».

Входное изображение будет выводиться в белом поле слева. Выходное изображение будет выводиться в отдельном окне.

В правой части видны входные параметры. Пользователю предоставлена возможность выбирать wavelet и уровень декомпозиции. Порог шумоподавления вводится в текстовом поле или при помощи

прокрутки. Возможен автоматический подбор порога. При этом ручной ввод становится недоступным.

Далее разрабатывается соответствующий код, выполняется отладка и тестирование. Листинг главного окна «Form1.cs» находится в приложении 3.

Также следует разработать окно для работы фильтров. Окно будет запускаться с параметрами. От них будет зависеть, какой выбран фильтр. Листинг окна фильтров «Filters.cs» находится в приложении И.

Таким образом, была разработана система, выполняющая очистку изображения, используя при этом wavelet-преобразования.

4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

4.1 Наложение шума на изображения

Первым шагом, возможно, наложение шума на изображение. Для этого в главном окне требуется нажать «Изображение» -> «Наложить шум». Запустится диалог выбора файла. Пользователю следует выбрать исходное изображение. Далее, откроется соответствующее окно. Оно изображено на рисунке 14.

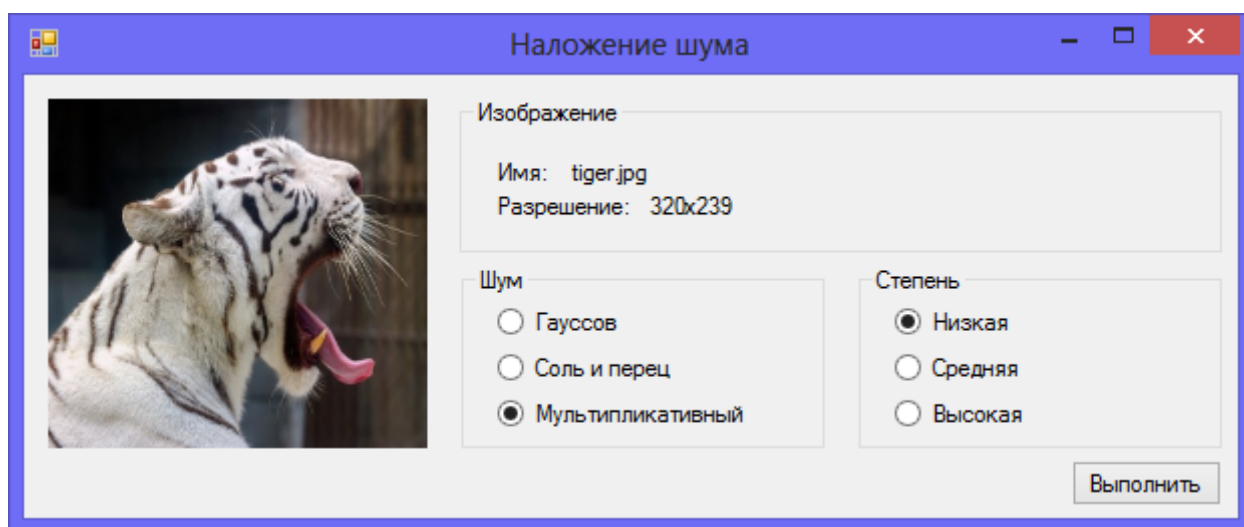


Рисунок 14 – Окно наложения шума

В этом окне пользователю предстоит выбрать шум и степень шума. Возможен выбор только одной позиции в каждой группе элементов управления.

В левой части выводится исходное изображение. Сверху видна информация об изображении – имя и разрешение.

Нажатие кнопки «Выполнить» приводит к наложению выбранного шума на изображение. Откроется диалог сохранения зашумленного изображения.

4.2 Фильтрация изображений при помощи wavelet-преобразований

Для открытия изображения следует нажать «Изображение» -> «Открыть». Появится диалог выбора файла. После загрузки изображения необходимо настроить параметры wavelet-преобразования: выбрать wavelet и уровень преобразования. Далее следует задать порог устранения шума. Эти действия выполнены и отражены на рисунке 15.

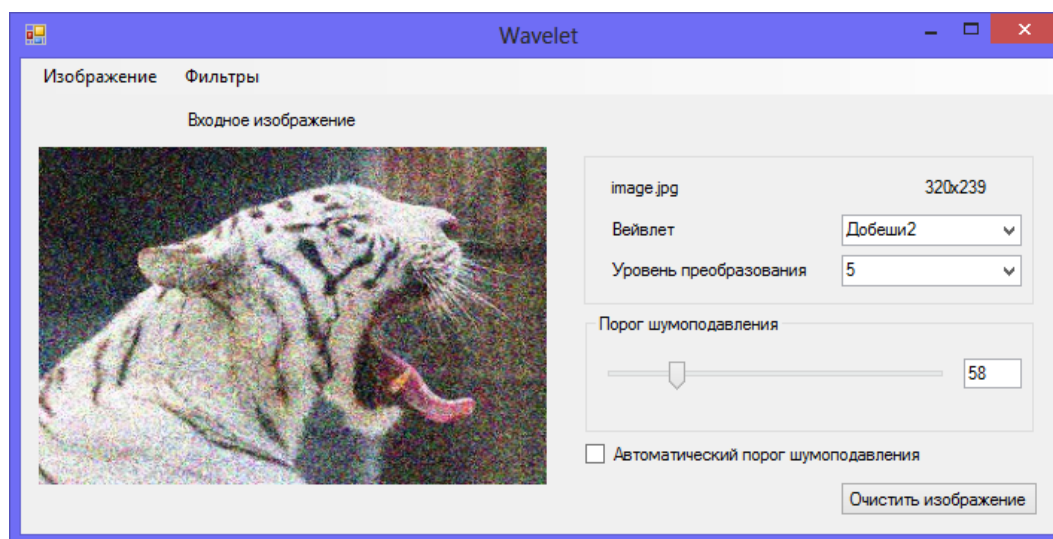


Рисунок 15 – Очистка изображения при помощи wavelet-декомпозиции

После нажатия кнопки «Очистить изображение» появится новое окно. В нем будет отображено исходное изображение и очищенное. Это отображено на рисунке 16.

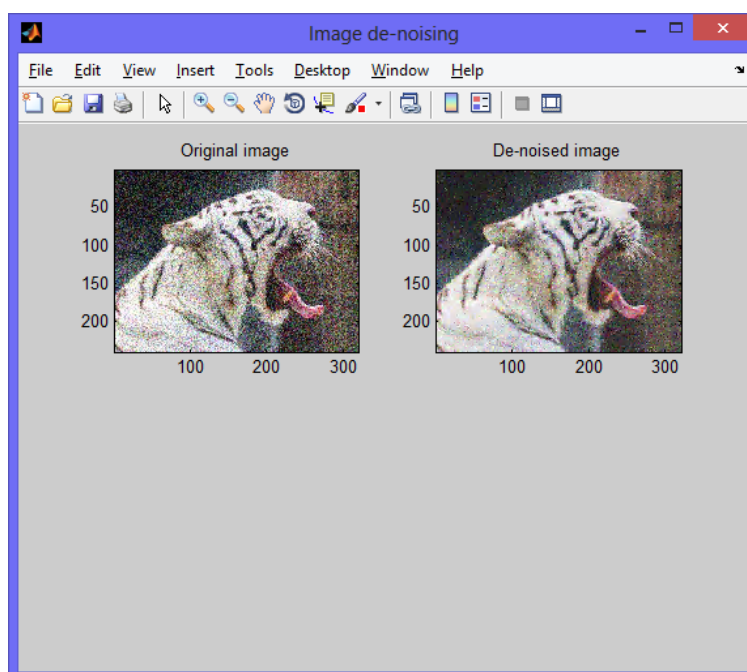


Рисунок 16 – Окно результата работы программы

Программа позволяет сохранять очищенное изображение путем нажатия «Изображение» -> «Сохранить очищенное». Откроется диалог сохранения файла. Следует ввести имя изображения и выбрать путь для него.

Возможен второй вариант работы. Он заключается в автоматическом выборе порога устранения шума. Для этого необходимо выполнить аналогичные действия, за исключением выбора порога. Вместо этого, следует выбрать «Автоматический порог шумоподавления». Это изображено на рисунке 17.

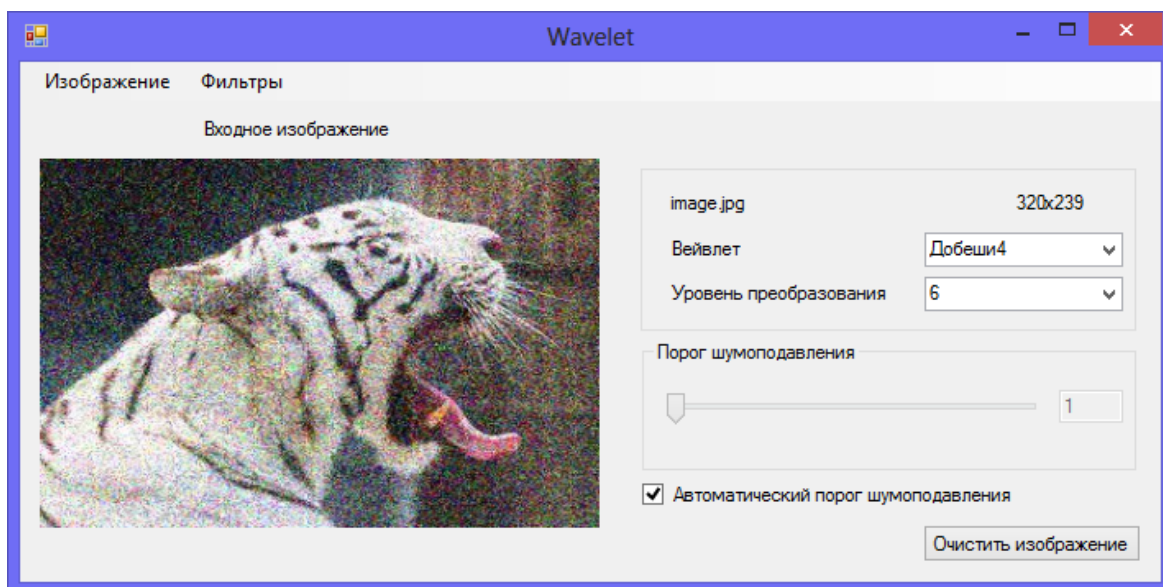


Рисунок 17 – Очистка изображения при помощи wavelet-декомпозиции с автоматическим порогом шумоподавления

Результаты работы будут выглядеть такие же, как и на рисунке 16.

4.3 Использование фильтров

Для фильтрации изображений могут использоваться фильтры. Они были описаны и разработаны на предыдущих этапах.

Для работы с фильтрами необходимо в главном окне загрузить изображение нажатием «Изображение» – «Открыть».

Далее следует нажать вкладку «Фильтры» и выбрать соответствующий фильтр. Работа со всеми фильтрами абсолютно одинакова. Разница заключается в их алгоритмической и программной составляющих.

Для примера приведено окно усредняющего фильтра (рисунок 18). Для работы фильтра необходимо выбрать размер окна (апертуры) и нажать очистить.

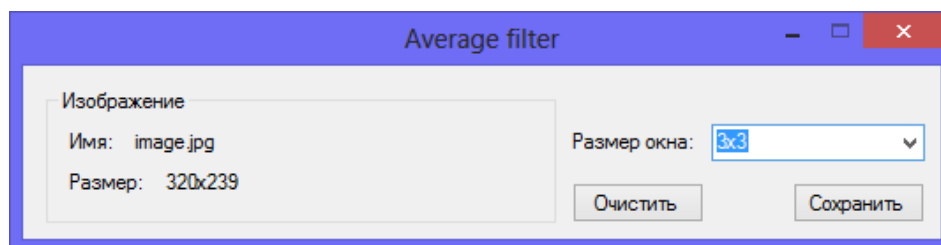


Рисунок 18 – Усредняющий фильтр

После нажатия кнопки «Очистить» появится результат работы программы. Он приведен на рисунке 19.

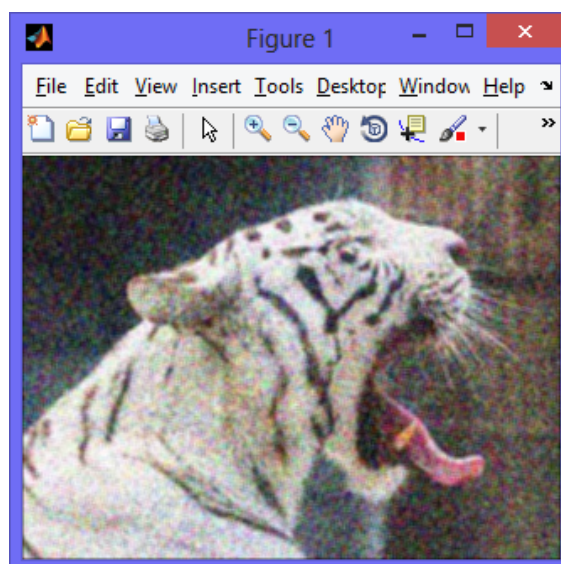


Рисунок 19 – Результат работы линейного (усредняющего) фильтра

После нажатия кнопки «Сохранить» появится диалоговое окно. Необходимо задать путь для сохранения изображения.

5 АНАЛИЗ РЕЗУЛЬТАТОВ

Основной результат работы системы – улучшение качества изображения. Данный показатель зависит от соотношения сигнал-шум, статистических и градационных характеристик шума, от спектра сигнала и т.д. Оценить качество изображения сложно, для этого необходимо использовать комплекс показателей. В данной работе мы применили субъективную сравнительную оценку. Сравнивали визуально три основных свойства изображения: контрастность, яркость и резкость. В зависимости от типа шума получили следующие результаты.

5.1 Гауссов шум

Результаты очистки входного изображения приведены в приложении К. В нем размещены исходное изображение, изображение, очищенное при помощи wavelet-преобразования и четыре изображения, очищенные усредняющим, Гауссовым, медианным и адаптивным фильтрами.

Для анализа результата фильтрации изображения, зашумленного Гауссовым шумом, описанными в работе способами, был наложен названный шум в средней степени. Результаты анализа приведены в приложении К.

Wavelet-фильтрация не ухудшила контрастность, снизила зашумленность, показав хороший видимый результат. Четкость изображения при этом не изменилась.

Усредняющий фильтр неплохо ослабил шум, но при этом сильно размыл изображение. Фильтр сохранил контрастность.

Фильтр Гаусса хорошо справился с данной задачей.

Результат работы медианного фильтра находится примерно между wavelet-преобразованием и усредняющим фильтром. Изображение было размыто, но не так сильно, как в случае усредняющего фильтра. Результат работы фильтра – достойное ослабление шума.

Адаптивный фильтр неплохо справился с задачей. Изображение не было размыто, но возник «холодный эффект» на изображении.

Таким образом, wavelet-преобразование не размывает изображение, как усредняющий и медианный фильтр. При фильтрации не возникает каких-либо побочных эффектов, как при адаптивной очистке. В целом, на изображении имеются остатки шума. Сильное повышение уровня фильтрации и порога шумоподавления может привести к размыванию изображения.

5.2 Шум «соль и перец»

Визуальный результат фильтрации изображения, зашумленного шумом «соль и перец» находится в приложении Л.

Wavelet-фильтрация заметно снизила уровень шума. Изображение было незначительно размыто. Контрастность осталась неизменной. В целом, показан хороший результат.

Усредняющий фильтр устранил шум, но, как и в предыдущем случае, значительно размыл изображение. Оно стало менее ярким.

Фильтр Гаусса не устранил шум. Он не предназначен для данного вида шума.

Медианный фильтр показал хороший результат, но при данной фильтрации изображение было заметно размыто. Были сглажены границы объектов (тигра и фона) на изображении.

Адаптивный фильтр не устранил данный шум, как и фильтр Гаусса.

Таким образом, для устранения шума вида «соль и перец» применимы wavelet-преобразования и медианный фильтр. Выбор метода, скорее всего, зависит от необходимых результатов. Если требуется более простое решение и менее четкое изображение, то применим медианный фильтр. В противном случае – следует использовать wavelet-преобразования. В целом, wavelet-преобразования не скрыли остатков шума. Скорее всего, требуется очень тщательный подбор параметров очистки шума.

5.3 Мультипликативный шум

Результат сопоставления методов представлен в приложении М.

Важно заметить, что при данном зашумлении были искажены цвета, и изображение стало более темным.

Wavelet-фильтрация очистила изображение, показав хороший результат. Остатков шума не было замечено. В некоторых местах имеется незначительное размытие. При этом в очищенном изображении цвета похожи на исходные, и само изображение похоже на исходное.

Усредняющий фильтр, фильтр Гаусса и адаптивный фильтры не очистили изображение.

Медианный фильтр немного сгладил шум, но исходные цвета не были восстановлены.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы были спроектированы и разработаны компоненты системы wavelet-преобразования изображений. Система предназначена для повышения качества изображений, а именно для их фильтрации.

На начальном этапе были определены виды шумов. Для более глубокого анализа предметной области, были реализованы фильтры, решающие данную задачу алгоритмическим способом.

Разработанная программа позволяет очистить изображение от шума и сохранить его. При этом имеется возможность очищать изображение различными способами, настраивая при этом параметры фильтрации. Результат будет зависеть от используемого способа. Это было продемонстрировано в предыдущем разделе.

Усредняющий фильтр неплохо ослабляет Гауссов шум, в небольшой степени фильтрует изображение от шума «соль и перец». Возникают затруднения при устранении мультипликативного шума. Если выбрать маленькое окно (апертуру), то усредняющий фильтр не устранил мультипликативный шум. Если окно будет большим, то фильтр устранил шум. Во всех случаях изображение получается размытым, что недопустимо в настоящее время.

Фильтр Гаусса предназначен для устранения одноименного шума. С этой задачей фильтр справляется. В остальных случаях метод неприменим.

Медианный фильтр показал хорошие результаты, особенно при устранении шума «соль и перец». С очисткой мультипликативного шума фильтр не справился. Если подобрать большое окно, то фильтр устранил мультипликативный шум, сильно размыв изображение.

Адаптивный фильтр устранил Гауссов шум. Фильтр не очищает изображение от шума «соль и перец». Возникают сложности с удалением мультипликативного шума. Если выбрать маленькое окно, то

мультипликативный шум не будет устранен. Если выбрать большое окно, то возникнет сильный эффект, похожий на взгляд на изображение через заледеневшее окно. В целом, было замечено, что данный эффект является проблемой адаптивного фильтра Винера.

Wavelet-преобразование устраняет все названные виды шума, показывая при этом хорошие результаты. Универсальность метода, несомненно, является его достоинством.

При wavelet-преобразовании возможно размытие изображения в случае выбора слишком высокого значения пороговой обработки и (или) уровня декомпозиции. При малых значениях этих параметров очищенное изображение будет слабо отличимым от зашумленного. Оба показателя прямо пропорционально влияют на результат. Таким образом, данный математический аппарат предоставляет гибкий набор параметров для устранения шума.

Wavelet-преобразования являются новым научным направлением. Их свойства и практическое применение полностью не рассмотрены. Несмотря на это, wavelet-преобразования уже используются в медицине, геофизике, радиотехнике, задачах распознавания образов и для сжатия информации. Таким образом, дальнейшее изучение данного научного направления является оправданным.

Разработанная система демонстрирует одно из возможностей wavelet-преобразований – фильтрацию изображений. Систему можно доработать, добавив функции сжатия изображений. Более существенное улучшение системы может заключаться в изменении ее реализации. Возможен отказ от использования системы компьютерной математики «Matlab», как стороннего объекта. В этом случае вариантом реализации является разработка dll-библиотек для wavelet-преобразований. Это возможно, как в среде «Matlab» (при этом не будет необходимости в использовании Matlab для работы программы), так при самостоятельной реализации. Последний вариант

является сложным и трудоемким. Для принятия такого решения необходимо учесть различные факторы.

Таким образом, требования технического задания выполнены полностью и в заданные сроки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Астафьева, Н.М. Вейвлет-анализ: основы теории и примеры применения [Текст] / Н.М. Астафьева // Успехи физических наук. – 2014. – №11. – С. 1145–1170.

2 Бондина, Н.Н. Сравнительный анализ алгоритмов фильтрации медицинских изображений [Текст] / Н.Н. Бондина, А.С. Калмычков, В.Э. Кривенцов // Вестник Национального технического университета Харьковский политехнический институт. Серия: Информатика и моделирование. – 2012. – №38 – С. 14–26.

3 Васильева, Л.Г. Преобразования Фурье и вейвлет-преобразования. Их свойства и применение [Текст] / Л.Г. Васильева, Я.М. Жилейкин, Ю.И. Осипик // Вычислительные методы и программирование. – 2011. – №3. – С. 172–175.

4 Грибунин, В.Г. Введение в вейвлет-преобразование [Электронный ресурс] / В.Г. Грибунин – Режим доступа: <http://www.autex.spb.su/download/wavelet/books/tutorial.pdf> – (Дата обращения: 16.05.2017).

5 Цифровая обработка изображений в информационных системах : учебное пособие [Текст] / И.С. Грузман [и др.] – Новосибирск.: Изд-во НГТУ, 2015. – 168 с.

6 Дьяконов, В.П. Matlab. Обработка сигналов и изображений : специальный справочник [Текст] / В.П. Дьяконов, И.В. Абраменкова. – СПб: Питер, 2012. – 608 с.

7 Московский, С.Б. Очистка сигнала от шумов с использованием вейвлет-преобразования [Электронный ресурс] / С.Б. Московский, А.Н. Сергеев, Н.А. Лалина // Universum: Технические науки : электрон. научн. журн. – 2015. – №2 – Режим доступа: <http://7universum.com/ru/tech/archive/item/1958> – (Дата обращения: 27.02.2017).

8 Смоленцев, Н.К. Основы теории вейвлетов. Вейвлеты в Matlab [Текст] / Н.К. Смоленцев. – 2-е изд. перераб. и доп. – М.: Изд-во ДМК, 2015. – 304 с.

9 Стругайло, В.В. Обзор методов фильтрации и сегментации цифровых изображений [Электронный ресурс] / В.В. Стругайло // Наука и образование: Электрон. науч.-техн. журн. – 2012. – №5. – Режим доступа: <http://cyberleninka.ru/article/n/obzor-metodov-filtratsii-i-segmentatsii-tsifrovyyh-izobrazheniy> – (Дата обращения 27.02.2017).

10 Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. : Пер. с англ. — М. : ООО «И.Д. Вильямс», 2013. — 1312 с. : ил. — Парал. тит. англ.

11 Системные требования для ОС Windows 7 [Электронный ресурс] : Официальная страница Microsoft – Режим доступа: <https://support.microsoft.com/ru-ru/help/10737/windows-7-system-requirements> – (Дата обращения: 16.05.2017).

12 Доля рынка операционной системы во всем мире [Электронный ресурс] : Глобальная статистика StatCounter – Режим доступа: <http://gs.statcounter.com/os-market-share> – (Дата обращения: 16.05.2017).

Приложение А

Класс wavelet-преобразования – «WaveletClass.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MApp;
using System.Drawing;
using System.Windows.Forms;

namespace Wavelet
{
    public class WaveletClass
    {
        //COM-объект для работы с Matlab
        MAppClass MyMatlab = new MAppClass();

        public bool isClear = false;

        //Очистка workspace
        public void ClearWorkSpace()
        {
            MyMatlab.Execute("clear all;");
        }

        //Загрузка изображения
        public void OpenImage(string ImageWay)
        {
            string way = "X = imread('";
            way += ImageWay;
            way += "')";

            MyMatlab.Execute(way);

            //Новое изображение не было очищено
            isClear = false;
        }

        //Вейвлет-очистка изображения
        public void DeNoising(string WaveName, int Level, int
Threshold)
        {
            //Имя вейвлета
            string wname = "w = '";
            wname += WaveName;
            wname += "'";
            MyMatlab.Execute(wname);
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

68

```

//Уровень преобразования
string level = "n = ";
level += Level.ToString();
level += ";";
MyMatlab.Execute(level);

//Значение порога преобразования
string threshold = "thr = ";
threshold += Threshold.ToString();
threshold += ";";
MyMatlab.Execute(threshold);

//Вейвлет-декомпозиция
MyMatlab.Execute("[c, l] = wavedec2(X, n, w);");

//Очистка изображения
MyMatlab.Execute("xd = wdencomp('gbl', c, l, w, n, thr,
'h', 1);");

MyMatlab.Execute("xd = round(xd);");
MyMatlab.Execute("if isequal(class(X), 'uint8'), xd =
uint8(xd); end");

//Вывод изображений
MyMatlab.Execute("figure('Name','Image de-
noising','NumberTitle','off')");
MyMatlab.Execute("subplot(221), image(X), title('Original
image')");
MyMatlab.Execute("subplot(222), image(xd), title('De-
noised image')");

this.isClear = true;
}

//Очистка с автоматическим выбором порога
public void DeNoisingAutoThrsh(string Wavelet, int Level)
{
//Имя вейвлета
string wname = "w = ";
wname += Wavelet;
wname += ";";
MyMatlab.Execute(wname);

//Уровень преобразования
string level = "n = ";
level += Level.ToString();
level += ";";
MyMatlab.Execute(level);

```

```

//Вейвлет-преобразование
MyMatlab.Execute("[c, s] = wavedec2(X, n, w);");

MyMatlab.Execute("alpha = 3;");
MyMatlab.Execute("m = 2.7*prod(s(1,:));");

//Получение порога очистки шума
MyMatlab.Execute("thr = wdcbm2(c, s, alpha, m);");

//Очистка шума
MyMatlab.Execute("xd = wdencomp('lvd', X, w, n, thr,
'h');");

MyMatlab.Execute("xd = round(xd);");
MyMatlab.Execute("if isequal(class(X), 'uint8'), xd =
uint8(xd); end");

//Имя окна
MyMatlab.Execute("figure('Name','Image de-
noising','NumberTitle','off')");

//Вывод результатов
MyMatlab.Execute("subplot(221), image(X), title('Original
image')");
MyMatlab.Execute("subplot(222), image(xd), title('De-
noised image')");

//Изображение очищено
this.isClear = true;
}

//Сохранение очищенного изображения
public void SaveClear(string SaveFile)
{
    //Если изображение было очищено
    if (isClear)
    {
        string m_save = "imwrite(xd, '";
        m_save += SaveFile;
        m_save += "')";
        try
        {
            MyMatlab.Execute(m_save);
        }
        catch
        {
            MessageBox.Show("Невозможно сохранить
изображение", "Ошибка", MessageBoxButtons.OK);
            return;
        }
    }
}

```

```
    }
    else
    {
        MessageBox.Show("Изображение не было очищено.",
"Ошибка", MessageBoxButtons.OK);
        return;
    }
}
}
```

Приложение Б

Класс наложения шума – «Noise.cs»

```
using MApp;
using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public class Noise
    {
        //COM-объект для работы с matlab
        MAppClass MyMatlab = new MAppClass();

        //Гауссов шум
        public void Gaussian(string rate, string OpenFile, string
SaveFile)
        {
            //Открыть изображение в matlab
            string m_file = "N = imread('";
            m_file += OpenFile;
            m_file += "')";
            MyMatlab.Execute(m_file);

            //Зашумление изображения
            //слабо
            if (rate == "low")
            {
                MyMatlab.Execute("I = imnoise(N, 'gaussian', 0,
0.02);");
            }
            //средне
            if (rate == "medium")
            {
                MyMatlab.Execute("I = imnoise(N, 'gaussian', 0.1,
0.04);");
            }
            //сильно
            if (rate == "high")
            {
                MyMatlab.Execute("I = imnoise(N, 'gaussian', 0.3,
0.05);");
            }
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

72

```

        //Сохранение зашумленного изображения
        string m_save = "imwrite(I, '";
        m_save += SaveFile;
        m_save += "');";
        try
        {
            MyMatlab.Execute(m_save);
        }
        catch
        {
            MessageBox.Show("Невозможно сохранить изображение",
"Ошибка", MessageBoxButtons.OK);
            return;
        }
    }

    //Шум соль и перец
    public void SaltAndPepper(string rate, string OpenFile, string
SaveFile)
    {
        //Открыть изображение в matlab
        string m_file = "N = imread('";
        m_file += OpenFile;
        m_file += "');";
        MyMatlab.Execute(m_file);

        //зашумление изображения
        //слабо
        if (rate == "low")
        {
            MyMatlab.Execute("I = imnoise(N, 'salt & pepper',
0.02);");
        }
        //средне
        if (rate == "medium")
        {
            MyMatlab.Execute("I = imnoise(N, 'salt & pepper',
0.07);");
        }
        //сильно
        if (rate == "high")
        {
            MyMatlab.Execute("I = imnoise(N, 'salt & pepper',
0.14);");
        }

        //Сохранение зашумленного изображения
        string m_save = "imwrite(I, '";
        m_save += SaveFile;
        m_save += "');";
    }

```

```

        try
        {
            MyMatlab.Execute(m_save);
        }
        catch
        {
            MessageBox.Show("Невозможно сохранить изображение",
"Ошибка", MessageBoxButtons.OK);
            return;
        }
    }

    //Мультипликативный шум
    public void Speckle(string rate, string OpenFile, string
SaveFile)
    {
        //Открыть изображение в matlab
        string m_file = "N = imread('";
        m_file += OpenFile;
        m_file += "')";
        MyMatlab.Execute(m_file);

        //Зашумление изображения
        //слабо
        if (rate == "low")
        {
            MyMatlab.Execute("I = imnoise(N, 'speckle', 0.02);");
        }
        //средне
        if (rate == "medium")
        {
            MyMatlab.Execute("I = imnoise(N, 'speckle', 0.07);");
        }
        //сильно
        if (rate == "high")
        {
            MyMatlab.Execute("I = imnoise(N, 'speckle', 0.14);");
        }

        //Сохранение зашумленного изображения
        string m_save = "imwrite(I, '";
        m_save += SaveFile;
        m_save += "')";
        try
        {
            MyMatlab.Execute(m_save);
        }
        catch
        {

```

```
        MessageBox.Show("Невозможно сохранить изображение",  
"Ошибка", MessageBoxButtons.OK);  
        return;  
    }  
}  
}
```

Приложение В

Форма наложения шума – «NoiseForm.cs»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public partial class NoiseForm : Form
    {
        Noise MyNoise = new Noise();

        public NoiseForm()
        {
            InitializeComponent();
            pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
            radioButton1.Checked = true;
            radioButton4.Checked = true;

            //Фильтр для входных файлов
            openFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg|Bitmap
Image (*.bmp)|*.bmp|Png Image (*.png)|*.png|All files (*.*)|*.*";
            openFileDialog1.FileName = string.Empty;

            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                Bitmap noisePic;
                try
                {
                    noisePic = new Bitmap(openFileDialog1.FileName);
                    pictureBox1.Image = (Image)noisePic;
                }
                catch
                {
                    MessageBox.Show("Невозможно открыть изображение",
"Ошибка", MessageBoxButtons.OK);
                    return;
                }
                //Имя и разрешение изображения
                label2.Text =
Path.GetFileName(openFileDialog1.FileName);
            }
        }
    }
}
```

```

        label4.Text = noisePic.Width.ToString() + "x" +
noisePic.Height.ToString();
    }
    else
    {
        this.Close();
    }
}

//Зашумить изображение
private void button1_Click(object sender, EventArgs e)
{
    saveFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg|Bitmap
Image (*.bmp)|*.bmp|Png Image (*.png)|*.png|All files (*.*)|*.*";
    saveFileDialog1.DefaultExt = "jpg";
    saveFileDialog1.FileName = "image";
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        //Выбор вида шума
        //Гауссов
        if (radioButton1.Checked)
        {
            if (radioButton4.Checked)
            {
                MyNoise.Gaussian("low",
openFileDialog1.FileName, saveFileDialog1.FileName);
            }
            if (radioButton5.Checked)
            {
                MyNoise.Gaussian("medium",
openFileDialog1.FileName, saveFileDialog1.FileName);
            }
            if (radioButton6.Checked)
            {
                MyNoise.Gaussian("high",
openFileDialog1.FileName, saveFileDialog1.FileName);
            }
        }
        //Соль и перец
        if (radioButton2.Checked)
        {
            if (radioButton4.Checked)
            {
                MyNoise.SaltAndPepper("low",
openFileDialog1.FileName, saveFileDialog1.FileName);
            }
            if (radioButton5.Checked)
            {
                MyNoise.SaltAndPepper("medium",
openFileDialog1.FileName, saveFileDialog1.FileName);
            }
        }
    }
}

```

```
        if (radioButton6.Checked)
        {
            MyNoise.SaltAndPepper("high",
openFileDialog1.FileName, saveFileDialog1.FileName);
        }
    }

    //Мультипликативный
    if (radioButton3.Checked)
    {
        if (radioButton4.Checked)
        {
            MyNoise.Speckle("low",
openFileDialog1.FileName, saveFileDialog1.FileName);
        }
        if (radioButton5.Checked)
        {
            MyNoise.Speckle("medium",
openFileDialog1.FileName, saveFileDialog1.FileName);
        }
        if (radioButton6.Checked)
        {
            MyNoise.Speckle("high",
openFileDialog1.FileName, saveFileDialog1.FileName);
        }
    }
}
else
{
    return;
}
}
```

Приложение Г

Абстрактный класс фильтра – «Filter.cs»

```
using MApp;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public abstract class Filter
    {
        //COM-объект матлаб
        protected MAppClass MyMatlab = new MAppClass();

        //Было ли очищено изображение
        public bool isClear;

        //Конструктор
        public Filter()
        {
            this.isClear = false;
        }

        //Размер окна
        protected string m_window;

        //Выставить окно для фильтров
        public void Clear(string window)
        {
            switch (window)
            {
                case "2x2":
                    m_window = "[2 2]";
                    break;
                case "3x3":
                    m_window = "[3 3]";
                    break;
                case "4x4":
                    m_window = "[4 4]";
                    break;
                case "5x5":
                    m_window = "[5 5]";
                    break;
                case "6x6":
                    m_window = "[6 6]";
                    break;
            }
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

79

```

        case "7x7":
            m_window = "[7 7]";
            break;
        case "8x8":
            m_window = "[8 8]";
            break;
        case "9x9":
            m_window = "[9 9]";
            break;
        default:
            return;
    }

    //Окно по размеру изображения
    MyMatlab.Execute("iptsetpref('ImshowBorder','tight')");
}

//Сохранить очищенное изображение
public void SaveClearImage(string FileSave)
{
    string m_save = "imwrite(i, '";
    m_save += FileSave;
    m_save += "')";
    try
    {
        MyMatlab.Execute(m_save);
    }
    catch
    {
        MessageBox.Show("Невозможно сохранить изображение",
            "Ошибка", MessageBoxButtons.OK);
        return;
    }
}
}
}
}

```

Приложение Д

Класс линейного фильтра – «LinearFilter.cs»

```
using MApp;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public class LinearFilter : Filter
    {
        //Наследование конструктора
        public LinearFilter()
            : base() { }

        //Очитка изображения
        public void Clear(string type, string FileName, string window)
        {
            base.Clear(window);

            //Открыть изображение
            string m_name = "X = imread('";
            m_name += FileName;
            m_name += "')";
            try
            {
                MyMatlab.Execute(m_name);
            }
            catch
            {
                MessageBox.Show("Невозможно открыть изображение",
"Ошибка", MessageBoxButtons.OK);
                return;
            }

            //Сформировать маску фильтра
            string filter = " h = fspecial('";
            filter += type;
            filter += "', ";
            filter += m_window;
            filter += "');";
            MyMatlab.Execute(filter);

            //Фильтрация изображения
            MyMatlab.Execute("i = imfilter(X, h);");
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

81

```

        //Вывод изображения
        MyMatlab.Execute("imshow(i)");

        //Очистка выполнена
        isClear = true;
    }

    //Сохранить очищенное изображение
    public void SaveClearImage(string FileSave)
    {
        base.SaveClearImage(FileSave);
    }
}

```

Приложение Е

Класс медианного фильтра – «Median Filter.cs»

```
using MApp;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public class MedianFilter : Filter
    {
        //Конструктор
        public MedianFilter() : base() { }

        //Очистка изображения
        public void Clear(string FileName, string Window)
        {
            base.Clear(Window);

            //Загрузка изображения
            string m_path = "X = imread('";
            m_path += FileName;
            m_path += "')";
            try
            {
                MyMatlab.Execute(m_path);
            }
            catch
            {
                MessageBox.Show("Невозможно открыть изображение",
"Ошибка", MessageBoxButtons.OK);
                return;
            }

            //Выделение цветовых составляющих
            MyMatlab.Execute("R = X(:,:,1);");
            MyMatlab.Execute("G = X(:,:,2);");
            MyMatlab.Execute("B = X(:,:,3);");

            //Очистка изображения
            MyMatlab.Execute("i(:,:,1) = medfilt2(R, " + m_window +
");");
            MyMatlab.Execute("i(:,:,2) = medfilt2(G, " + m_window +
");");
            MyMatlab.Execute("i(:,:,3) = medfilt2(B, " + m_window +
");");
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

83

```

        //Вывести изображение
        MyMatlab.Execute("imshow(i)");

        this.isClear = true;
    }

    //Сохранить очищенное изображение
    public void SaveClearImage(string FileName)
    {
        base.SaveClearImage(FileName);
    }
}

```

Приложение Ж

Класс Адаптивного фильтра Винера – «AdaptiveFilter.cs»

```
using MApp;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public class AdaptiveFilter : Filter
    {
        public AdaptiveFilter() : base() { }

        //Фильтрация изображения
        public void Clear(string FileName, string Window)
        {
            base.Clear(Window);

            //Загрузка изображения
            string m_path = "X = imread('";
            m_path += FileName;
            m_path += "')";
            try
            {
                MyMatlab.Execute(m_path);
            }
            catch
            {
                MessageBox.Show("Невозможно открыть изображение",
                "Ошибка", MessageBoxButtons.OK);
                return;
            }

            //Выделение цветовых составляющих
            MyMatlab.Execute("R = X(:, :, 1);");
            MyMatlab.Execute("G = X(:, :, 2);");
            MyMatlab.Execute("B = X(:, :, 3);");

            //Очистка изображения
            MyMatlab.Execute("i(:, :, 1) = wiener2(R, " + m_window +
            ");");
            MyMatlab.Execute("i(:, :, 2) = wiener2(G, " + m_window +
            ");");
            MyMatlab.Execute("i(:, :, 3) = wiener2(B, " + m_window +
            ");");
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

85

```

        //Вывести изображение
        MyMatlab.Execute("imshow(i)");

        this.isClear = true;
    }

    //Сохранить очищенное изображение
    public void SaveClearImage(string FileName)
    {
        base.SaveClearImage(FileName);
    }
}

```

Приложение 3

Листинг главного окна – «Form1.cs»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public partial class Form1 : Form
    {
        WaveletClass MyWavelet = new WaveletClass();

        //Уровень пороговой обработки шума
        private const int maxLevel = 300;
        private const int minLevel = 1;

        public Form1()
        {
            InitializeComponent();

            //Доступные вейвлеты
            comboBox1.Items.Add("Хаара");
            comboBox1.Items.Add("Добеши2");
            comboBox1.Items.Add("Добеши4");
            comboBox1.Items.Add("Симлет2");
            comboBox1.Items.Add("Симлет4");
            comboBox1.Items.Add("Койфлет2");
            comboBox1.Items.Add("Койфлет4");
            comboBox1.SelectedItem = "Хаара";

            //Уровни преобразования
            for (int i = 1; i < 11; i++)
            {
                comboBox2.Items.Add(i.ToString());
            }
            comboBox2.SelectedItem = "1";

            //Порог обработки шума
            trackBar1.Minimum = minLevel;
            trackBar1.Maximum = maxLevel;
            trackBar1.TickFrequency = minLevel;
            trackBar1.Value = minLevel;
        }
    }
}
```

```

        textBox1.Text = trackBar1.Value.ToString();
        trackBar1.TickStyle = TickStyle.None;
    }

    private void trackBar1_ValueChanged(object sender, EventArgs
e)
    {
        textBox1.Text = trackBar1.Value.ToString();
    }

    private void textBox1_TextChanged(object sender, EventArgs e)
    {
        try
        {
            if (textBox1.Text == string.Empty)
            {
                return;
            }

            if (int.Parse(textBox1.Text) >= 1 &&
(int.Parse(textBox1.Text) <= maxLevel))
            {
                trackBar1.Value = int.Parse(textBox1.Text);
            }
        }
        catch (Exception e1)
        {
            MessageBox.Show(e1.Message);
        }
    }

    private void OpenToolStripMenuItem_Click(object sender,
EventArgs e)
    {
        openFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg|Bitmap
Image (*.bmp)|*.bmp|Png Image (*.png)|*.png|All files (*.*)|*.*";
        openFileDialog1.FileName = string.Empty;
        if (openFileDialog1.ShowDialog() == DialogResult.OK)
        {
            //Загрузить изображение
            Bitmap mypic;

            try
            {
                mypic = new Bitmap(openFileDialog1.FileName);
                //Размеры picturebox'a
                pictureBox1.SizeMode =
PictureBoxSizeMode.StretchImage;

```

```

        //Вывести изображение
        pictureBox1.Image = (Image)mypic;
    }
    catch
    {
        MessageBox.Show("Невозможно открыть изображение",
"Ошибка", MessageBoxButtons.OK);
        return;
    }

    //Вывести имя изображения и размеры
    label3.Text =
Path.GetFileName(openFileDialog1.FileName);
    label4.Text = mypic.Width.ToString();
    label4.Text += "x";
    label4.Text += mypic.Height.ToString();
}
else
{
    return;
}
}

private void button2_Click(object sender, EventArgs e)
{
    //очистить workspace
    //Загрузить изображение в Matlab
    if (openFileDialog1.FileName != string.Empty)
    {
        MyWavelet.ClearWorkSpace();
        MyWavelet.OpenImage(openFileDialog1.FileName);
    }

    //Выбрать Wavelet
    string wname = "";
    switch (comboBox1.Text)
    {
        case "Хаара":
            wname = "haar";
            break;

        case "Добеши2":
            wname = "db2";
            break;

        case "Добеши4":
            wname = "db4";
            break;
    }
}

```

```

        case "Симлет2":
            wname = "sym2";
            break;

        case "Симлет4":
            wname = "sym4";
            break;

        case "Койфлет2":
            wname = "coif2";
            break;

        case "Койфлет4":
            wname = "coif4";
            break;
    }

    //Уровень вейвлет-преобразования
    int level = comboBox2.SelectedIndex + 1;

    //Порог обработки
    int thrshold = int.Parse(textBox1.Text);

    //Очистка шума
    if (checkBox1.Checked)
    {
        //Очистка с автоматическим определением порога
        шумоподавления
        MyWavelet.DeNoisingAutoThrsh(wname, level);
    }
    else
    {
        //Очистка с указанным порогом
        MyWavelet.DeNoising(wname, level, thrshold);
    }
}

private void CloseToolStripMenuItem_Click(object sender,
EventArgs e)
{
    //Очистить picturebox
    pictureBox1.Image = null;
    pictureBox1.Invalidate();
    //Исходные значения контролов
    comboBox1.SelectedIndex = 0;
    comboBox2.SelectedIndex = 0;
    trackBar1.Value = minLevel;
    //очистить имя файла и разрешение
    label3.Text = string.Empty;
    label4.Text = string.Empty;
}

```

```

        private void NoiseToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            //Создать и вывести форму
            Form NoiseForm = new NoiseForm();
            try
            {
                NoiseForm.Show();
            }
            catch
            {
                return;
            }
        }

        private void SaveClearedToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            //Если изображение не было очищено
            if (!MyWavelet.isClear)
            {
                MessageBox.Show("Нет изображения для сохранения.",
"Ошибка!", MessageBoxButtons.OK);
                return;
            }

            saveFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg|Bitmap
Image (*.bmp)|*.bmp|Png Image (*.png)|*.png|All files (*.*)|*.*";
            saveFileDialog1.DefaultExt = "jpg";
            saveFileDialog1.FileName = "clear image";
            if (saveFileDialog1.ShowDialog() == DialogResult.OK)
            {
                MyWavelet.SaveClear(saveFileDialog1.FileName);
            }
            else
            {
                return;
            }
        }

        //Усредняющий фильтр
        private void meanToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            //Открыть новую форму, если было открыто изображение
            if (pictureBox1.Image != null)
            {
                Form Linear = new Filters("average",
openFileDialog1.FileName, label4.Text);

```

```

        Linear.Text = "Average filter";
        Linear.Show();
    }
    else
    {
        MessageBox.Show("Откройте изображение.", "Внимание!",
        MessageBoxButtons.OK);
        return;
    }
}

//Фильтр гаусса
private void GaussFilterToolStripMenuItem_Click(object sender,
EventArgs e)
{
    //Открыть новую форму, если было открыто изображение
    if (pictureBox1.Image != null)
    {
        Form Linear = new Filters("gaussian",
openFileDialog1.FileName, label4.Text);
        Linear.Text = "Gaussian filter";
        Linear.Show();
    }
    else
    {
        MessageBox.Show("Откройте изображение.", "Внимание!",
        MessageBoxButtons.OK);
        return;
    }
}

//Медианный фильтр
private void medianToolStripMenuItem_Click(object sender,
EventArgs e)
{
    //Открыть новую форму, если было открыто изображение
    if (pictureBox1.Image != null)
    {
        Form Median = new Filters(true, false,
openFileDialog1.FileName, label4.Text);
        Median.Text = "Median filter";
        Median.Show();
    }
    else
    {
        MessageBox.Show("Откройте изображение.", "Внимание!",
        MessageBoxButtons.OK);
        return;
    }
}
}

```

```

        //Адаптивный фильтр
        private void adaptiveToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            //Открыть новую форму, если было открыто изображение
            if (pictureBox1.Image != null)
            {
                Form Adaptive = new Filters(false, true,
openFileDialog1.FileName, label4.Text);
                Adaptive.Text = "Adaptive filter";
                Adaptive.Show();
            }
            else
            {
                MessageBox.Show("Откройте изображение.", "Внимание!",
MessageBoxButtons.OK);
                return;
            }
        }

        private void checkBox1_CheckedChanged(object sender, EventArgs
e)
        {
            if (checkBox1.Checked)
            {
                trackBar1.Enabled = false;
                textBox1.Enabled = false;
            }
            else
            {
                trackBar1.Enabled = true;
                textBox1.Enabled = true;
            }
        }
    }
}

```

Приложение И

Окно фильтров – «Filters.cs»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Wavelet
{
    public partial class Filters : Form
    {
        //Линейный фильтр
        LinearFilter MyLinearFilter;
        bool m_linear = false;

        //Медианный фильтр
        MedianFilter MyMedianFilter;
        bool m_median = false;

        //Адаптивный фильтр
        AdaptiveFilter MyAdaptiveFilter;
        bool m_adaptive = false;

        //Путь к файлу
        string m_file;

        //Тип фильтра
        string filter_type;

        public Filters(bool median, bool adaptive, string fname,
            string fsize)
        {
            InitializeComponent();

            //Имя и размеры изображения
            label3.Text = Path.GetFileName(fname);
            label4.Text = fsize;

            //Варианты размеров скользящего окна
            comboBox1.Items.Add("2x2");
            comboBox1.Items.Add("3x3");
            comboBox1.Items.Add("4x4");
            comboBox1.Items.Add("5x5");
        }
    }
}
```

Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

94

```

comboBox1.Items.Add("6x6");
comboBox1.Items.Add("7x7");
comboBox1.Items.Add("8x8");
comboBox1.Items.Add("9x9");
comboBox1.SelectedIndex = 0;

//Путь к файлу
m_file = fname;

if (median)
{
    MyMedianFilter = new MedianFilter();
    m_median = true;
}
if (adaptive)
{
    MyAdaptiveFilter = new AdaptiveFilter();
    m_adaptive = true;
}
}

public Filters(string filterType, string fname, string fsize)
{
    InitializeComponent();

    //Имя и размеры изображения
    label3.Text = Path.GetFileName(fname);
    label4.Text = fsize;

    //Варианты размеров скользящего окна
    comboBox1.Items.Add("2x2");
    comboBox1.Items.Add("3x3");
    comboBox1.Items.Add("4x4");
    comboBox1.Items.Add("5x5");
    comboBox1.Items.Add("6x6");
    comboBox1.Items.Add("7x7");
    comboBox1.Items.Add("8x8");
    comboBox1.Items.Add("9x9");
    comboBox1.SelectedIndex = 0;

    //Путь к файлу
    m_file = fname;

    //Тип фильтра
    filter_type = filterType;

    MyLinearFilter = new LinearFilter();
    m_linear = true;
}

```

```

//Фильтровать изображение
private void button1_Click(object sender, EventArgs e)
{
    if (m_linear)
    {
        MyLinearFilter.Clear(filter_type, m_file,
comboBox1.SelectedItem.ToString());
    }
    if(m_median)
    {
        MyMedianFilter.Clear(m_file,
comboBox1.SelectedItem.ToString());
    }
    if (m_adaptive)
    {
        MyAdaptiveFilter.Clear(m_file,
comboBox1.SelectedItem.ToString());
    }
}

//Сохранить очищенное изображение
private void button2_Click(object sender, EventArgs e)
{
    //Если изображение не было очищено
    if (((m_linear) && (!MyLinearFilter.isClear)) ||
((m_median) && (!MyMedianFilter.isClear))
|| ((m_adaptive) && (!MyAdaptiveFilter.isClear)))
    {
        MessageBox.Show("Изображение не было очищено.",
"Ошибка!", MessageBoxButtons.OK);
        return;
    }

    //Фильтр выбора файлов
    saveFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg|Bitmap
Image (*.bmp)|*.bmp|Png Image (*.png)|*.png|All files (*.*)|*.*";
    saveFileDialog1.DefaultExt = "jpg";
    saveFileDialog1.FileName = "clear image";

    //Сохранение изображения
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        if (m_linear)
        {
            MyLinearFilter.SaveClearImage(saveFileDialog1.FileName);
        }
        if (m_median)
        {
            MyMedianFilter.SaveClearImage(saveFileDialog1.FileName);
        }
    }
}

```

```

        }
        if (m_adaptive)
        {
MyAdaptiveFilter.SaveClearImage(saveFileDialog1.FileName);
        }
        }
    else
    {
        return;
    }
    }
}
}

```

Приложение К

Результаты очистки Гауссова шума

Исходное изображение	Зашумленное изображение
	
Wavelet-фильтрация	Усредняющий фильтр
	
Фильтр Гаусса	Медианный фильтр
	

Адаптивный фильтр



Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

99

Приложение Л

Результаты очистки шума «соль и перец»

Исходное изображение	Зашумленное изображение
	
Wavelet-фильтрация	Усредняющий фильтр
	
Фильтр Гаусса	Медианный фильтр
	

Адаптивный фильтр



Изм.	Лист	№ докум	Подпись	Дата

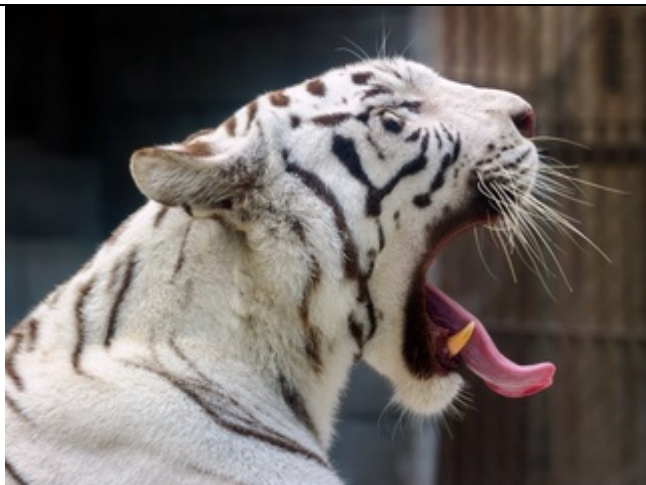
ВКР 090301.10. 17 81

Лист

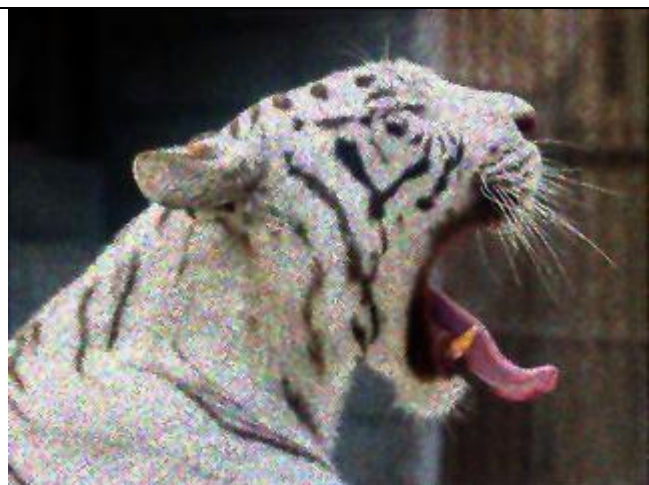
101

Приложение М

Результаты очистки мультипликативного шума

Исходное изображение	Зашумленное изображение
	
Wavelet-фильтрация	Усредняющий фильтр
	
Фильтр Гаусса	Медианный фильтр
	

Адаптивный фильтр



Изм.	Лист	№ докум	Подпись	Дата

ВКР 090301.10. 17 81

Лист

103