Санкт-Петербургский государственный университет

**Факультет прикладной математики-процессов управления**

**Кафедра теории управления**

**Фаустов Богдан Андреевич**

**Магистерская диссертация**

**Обнаружение цифровых водяных знаков**

Направление 01.04.02

Прикладная математика и информатика

Магистерская программа

Методы прикладной математики и информатики в задачах управления

Научный руководитель,  
кандидат техн. наук,  
доцент  
Гришкин В. М.

Рецензент,

кандидат техн. наук,  
Космачев В. М.

Санкт-Петербург — 2018

# Содержание

Введение 3

Глава 1. Цифровые водяные знаки 7

* + 1. Типы цифровых водяных знаков. 7
  1. Методы встраивания информации в изображение 12

Глава 2. Метод обнаружения прозрачных водяных знаков 16

* + 1. Описание метода 16
    2. Применение метода в системе информационного поиска. 20

Заключение 28

Список литературы 29

Приложение 31

**Введение**

Цифровой водяной знак (ЦВЗ) - это технология для предотвращения похищения или использования цифровых изображений, аудио и видео без разрешения владельца. ЦВЗ представляет собой внедрение цифровой подписи в данные. Существует два класса цифровых водяных знаков - видимые и невидимые.

Видимый водяной знак лучше всего использовать для данных, зрительный образ которых не портится при добавлении цифровой подписи. Преимуществом таких водяных знаков является то, что данные защищены авторским правом и их полноценное использование становится невозможным без удаления цифрового водяного знака. Подобные меры защиты значительно упрощают споры по авторскому праву, поскольку наличие или факт удаления водяных знаков можно легко обнаружить.

Невидимый водяной знак используется, когда внешний вид данных не может быть изменен. Невидимый ЦВЗ – это специальная метка, встраиваемая в цифровой контент, называемый контейнером, с целью защиты авторских прав и подтверждения целостности документа. Преимущество такого типа водяных знаков состоит в том, что их нельзя легко обнаружить. Потенциальные нарушители могут использовать данные, не подозревая, что они содержат маркировку владельца.

В настоящее время разработано много различных методов встраивания невидимых ЦВЗ в изображения, например, метод LSB (Last Significant Bit), метод псевдослучайного интервала, метод псевдослучайной перестановки (выбора), метод блочного скрытия. Наиболее распространённым, но и наименее устойчивым к искажениям является метод замены битов младших разрядов, представляющих яркость/цвет пикселя, или так называемый LSB-метод. Суть метода LSB заключается в замене младших значащих битов в байтах изображения (контейнере), отвечающих за кодирование цвета, на биты скрываемого сообщения. Основными достоинствами данного метода являются: 1) тот факт, что человеческий глаз в большинстве случаев не способен заметить изменения в младших битах; 2) простота самого метода; 3) возможность скрывать в относительно небольших изображениях достаточно большие объемы информации. Основным недостатком метода LSB является его высокая чувствительность к искажениям контейнера [1]. В работе [1] также обозначается вопрос об определении подлинности документов, будь то физические документы (деньги, ценные и конфиденциальные документы) или электронные документы (сканированные копии, фотографии, электронный печатный текст). Это вызвано увеличением объема документооборота между организациями, а также развитием технологий обмена документами. Внедряемые водяные знаки можно использовать как для доказательства подлинности документа, так и для встраивания в них определенной информации.

В настоящее время при формировании ЦВЗ применяется принцип встраивания метки, которая представляет собой узкополосный сигнал, в широком диапазоне частот маркируемого изображения. Этот метод реализуется посредством двух различных алгоритмов. В одном алгоритме информация передается посредством фазовой модуляции «несущей», представляющей собой псевдослучайную последовательность чисел. В другом – весь диапазон частот делится на несколько субдиапазонов и передача производится между этими субдиапазонами. В отношении маркируемого изображения метку можно рассматривать как некоторый дополнительный шум. Но так как в изображении всегда присутствует шум, то его незначительное возрастание за счет добавления метки не приводит к заметному для зрения увеличению искажений. Кроме того, сигнал, представляющий метку, распространяется по всему изображению, благодаря чему достигается устойчивость к обрезке изображения [2]. Параллельный алгоритм обучения нейронной сети с машиной опорных векторов в качестве выходного слоя [3] позволяет автоматизировать процедуру формирования тренировочных наборов при создании систем распознавания изображений.

Важнейшее применение ЦВЗ нашли в системах защиты от копирования, которые стремятся предотвратить или удержать от несанкционированного копирования цифровых данных. Применяют ЦВЗ в стеганографии (способ передачи или хранения [информации](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D1%8F) с учётом сохранения в тайне самого факта такой передачи или хранения), когда стороны обмениваются секретными сообщениями, внедрёнными в цифровой сигнал. Используется как средство защиты документов с фотографиями — паспортов, водительских удостоверений, кредитных карт с фотографиями. Хотя некоторые форматы цифровых данных могут также нести в себе дополнительную информацию, называемую [метаданными](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%B0%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5), ЦВЗ отличаются тем, что информация «зашита» прямо в сигнал. Объекты мультимедиа в этом случае будут представлять собой контейнеры (носители) данных. Основное преимущество состоит в наличии условной зависимости между событием подмены объекта идентификации и наличии элемента защиты — скрытого водяного знака. Подмена объекта идентификации приведёт к выводу о подделке всего документа.

Видимые прозрачные водяные знаки менее изучены. Их обнаружение является весьма важной задачей.

**Постановка задачи.** В работе рассматривается задача поиска потенциальных водяных знаков, которая возникает при:

1. Классификации изображений.
2. Проверке соблюдения авторских прав.

Используется в:

1. Средствах массовой информации.
2. Социальных сетях.
3. Различных цифровых источниках информации.

**Цель работы.** Разработка метода и создание программы для поиска потенциальных видимых водяных знаков на изображениях.

Целью данной работы является:

1. Провести анализ поиска прозрачных областей для создания предположений о наличии водяных знаков на изображении.
2. Разработать метод для поиска потенциальных водяных знаков.
3. Реализовать в виде программы в среде Wolfram Mathematica 11.3 представленный алгоритм поиска потенциальных водяных знаков на изображении.

**Апробация результатов.** Результаты магистерской диссертации неоднократно докладывались на кафедре теории управления СПбГУ, на XLIX международной научной конференции аспирантов и студентов «Процессы управления и устойчивость» (Санкт-Петербург, 2018 г.), на конференциях «Актуальные проблемы инновационного развития ядерных технологий» в рамках научной сессии НИЯУ МИФИ (Северск, 2017 г., 2018 г.). По теме магистерской диссертации опубликована одна работа [1\*] в рецензируемом издании из списка ВАК, и еще одна находится в печати [2\*].

**Структура и объем работы.** Диссертационная работа состоит из введения, двух глав, заключения, списка литературы и приложения. Работа изложена на 37 страницах. Список литературы содержит 10 наименований.

# Глава 1. Цифровые водяные знаки

Данная глава посвящена обзору существующих типов цифровых водяных знаков, а также обзору методов встраивания информации в изображение.

**1.1. Типы цифровых водяных знаков**

Существует два класса цифровых водяных знаков - видимые и невидимые. Невидимый водяной знак используется, когда внешний вид данных не должен быть изменен (рисунок 1), например, текст может быть встроен в файл изображения, который должен оставаться неизменным и точным, как исходное изображение без знака [4].

 Рисунок 1. Применение невидимого цифрового водяного знака

Преимущество такого типа водяных знаков состоит в том, что их нельзя легко обнаружить, таким образом, потенциальные нарушители могут использовать данные, не подозревая, что они содержат маркировку владельца, с другой стороны отсутствие каких-либо обозначений авторства может повлечь за собой бесконтрольное копирование и распространение в сети. Поэтому в интернете часто можно встретить полупрозрачные логотипы и ссылки авторов на опубликованных изображениях/видео [5].

Видимый водяной знак лучше всего использовать для данных, зрительный образ которых не портится при добавлении цифровой подписи. Преимуществом таких водяных знаков является то, что данные защищены авторским правом и их полноценное использование становится невозможным без удаления цифрового водяного знака. Несмотря на то, что видимый водяной знак может быть убран с изображения с помощью программ для редактирования, такой процесс занимает много времени, т.к. не представляется возможным произвести подобное редактирование в автоматическом режиме, а также остаются заметные следы обработки. Подобные меры защиты значительно упрощают споры по авторскому праву, поскольку наличие водяных знаков или удаление водяных знаков можно легко обнаружить.

Защита данных от копирования с помощью видимых водяных знаков предполагает наличие заметного фирменного логотипа или символьного поля, представляющего из себя различные виды маркировок, таких как имя автора, ссылка на источник, название компании (рисунок 2).



Рисунок 2. Пример изображения с фирменным логотипом BBC News

Размещение подобных объектов на исходном изображении может сильно отвлекать внимание или даже исказить вид картинки, что зачастую считается недопустимым при публикации материалов в различных источниках информации, поэтому при создании материалов часто используют слабозаметные логотипы. Для решения этой задачи хорошо подходит прозрачность. С ее помощью можно эффективно нанести авторские знаки отличия (рисунок 3), при этом сохраняя оригинальное представление образов.



Рисунок 3. Пример изображения с прозрачным водяным знаком Bigstock

Явление прозрачности хорошо изучено в физике. Прозрачность  
среды — свойство вещества направленно пропускать свет. Простой пример прозрачного объекта (рисунок 4), который однозначно можно отличить от остального окружающего мира — цветное стекло.



Рисунок 4. Цветное стекло

В цифровых изображениях прозрачность может быть представлена как отдельная характеристика. При их создании к цветовой модели добавляется понятие альфа-смешения — техники создания эффекта полупрозрачности путём объединения исходного пикселя с пикселем, уже находящимся во фрейм-буфере. Отсюда берет свое название альфа-канал, также известный как маска-канал, простой способ объединить переходную прозрачность с изображением. Например, формат PNG позволяет использовать 254 или 65534 уровня частичной прозрачности. Все три типа PNG-изображений («TrueColor», «GrayScale» и индексированная палитра) могут содержать альфа-информацию, хотя обычно она применяется лишь с «TrueColor» изображениями. Вместо того, чтобы сохранять три байта для каждого пикселя (красный, зелёный и синий, RGB), сохраняются четыре: красный, зелёный, синий и альфа, таким образом получается RGBA.

В редакторе изображений размещение водяных знаков можно произвести с помощью наложения — объединения нескольких изображений. Уменьшая или увеличивая значение альфа-канала, можно получать различные результаты.

На примере (рисунок 5), созданном в редакторе слева, находится полностью непрозрачная часть телефона, справа на 80% прозрачная часть сквозь которую видно фон.



Рисунок 5. Прозрачность в редакторе изображений

С помощью метода наложения можно успешно размещать фирменные логотипы, которые символизируют авторское право на конкретный цифровой продукт. Однако не все типы изображений поддерживают альфа-канал. Часто используемый формат JPEG содержит только информацию о цветовых каналах. При публикации файлов в таком формате информация о прозрачности не сохраняется. Можно лишь визуально оценить степень прозрачности объектов и образов на изображении.

# 1.2. Методы встраивания информации в изображение

Цифровые водяные знаки разделяют на три вида:

* робастные (устойчивые),
* хрупкие,
* полухрупкие.

Робастные ЦВЗ устойчивы к любым воздействиям на них. Они обычно используются в системах защиты от копирования и идентификации. Хрупкие ЦВЗ изменяются или разрушаются при незначительном изменении контейнера, используются для проверки целостности электронных документов. Полухрупкие ЦВЗ устойчивы для одних воздействий и неустойчивы для других, используются обычно для обнаружения атаки на сигнал.

Существует множество различных методов встраивания ЦВЗ в документы [6]:

* методы скрытия данных в пространственной области,
* методы скрытия в частотной области,
* статистические методы,
* методы искажения,
* структурные методы.

Рассмотрим те методы, на основе которых имеются многочисленные разработки.

Преимуществом алгоритмов встраивания данных в пространственной области является то, что ЦВЗ внедряется в области исходного изображения за счет манипуляций яркостью или цветовыми составляющими, и нет необходимости выполнять громоздкие преобразования изображений.

Суть метода замены наименее значащего бита (Least Significant Bits – LSB) заключается во встраивании информации путем побитной записи в наименее значащие биты изображения. Разница между пустым и заполненным контейнерами должна быть не ощутима для органов восприятия человека. Достоинствами метода являются быстрота (один шаг – один бит) и возможность записи большого объема информации (до 1/8 от объема контейнера). Основным недостатком метода LSB является его высокая чувствительность к искажениям контейнера.

В отличие от метода LSB, в котором каждый бит скрываемого сообщения записывается в последовательно идущие младшие биты, метод псевдослучайного интервала позволяет осуществлять случайное распределение битов секретной информации по контейнеру. При этом расстояние между встроенными битами псевдослучайно. Данный метод эффективен в случае длины секретного сообщения много меньшей количества пикселей изображения и не эффективен в случае большого объема скрываемой информации. Кроме того, биты скрываемого сообщения в контейнере размещены в той же последовательности, что и в секретном сообщении. Во избежание этого недостатка можно использовать метод псевдослучайной перестановки.

Суть метода псевдослучайной перестановки состоит в следующем: генератор псевдослучайных чисел формирует последовательность индексов

, ,…, и сохраняет *к*-й бит сообщения в пикселе с индексом . Затем псевдослучайная функция перестановки располагает биты в сообщении случайным образом. Достоинством данного метода является то, что биты скрываемого сообщения расположены хаотически, что повышает уровень надежности. А недостаток метода состоит в следующем: если количество битов скрываемого сообщения ненамного меньше количества младших битов изображения, то велика вероятность пересечения индексов и, как следствие, наложение одного бита сообщения на другой.

К методам скрытия данных в пространственной области также относится метод блочного скрытия. Суть его заключается в следующем: оригинальное изображение разбивается на конечное число непересекающихся блоков произвольной конфигурации, для каждого из которых формируется бит четности. В каждом блоке производится скрытие одного секретного бита. Достоинствами данного метода являются: возможность модификации такого пикселя в блоке, изменение которого приводит к минимальному изменению статистики контейнера; влияние последствий встраивания секретных данных в контейнер можно уменьшить за счет увеличения размера блока. Основным недостатком данного метода является его неустойчивость к искажениям.

Отметим, что методы псевдослучайного интервала, псевдослучайной перестановки и блочного скрытия являются своего рода усложнением метода LSB.

Изображение, заданное в цифровом виде можно преобразовать, используя ортогональные преобразования, и встраивать ЦВЗ в образ, который иногда называют частотным представлением. Частотные методы встраивают ЦВЗ в частотную область изображения с использованием ортогональных преобразований файла-контейнера и перераспределением его энергии. После декомпозиции ЦВЗ встраивается в определенные спектральные коэффициенты контейнера. В результате применения соответствующего преобразования наибольшая энергия изображения сосредотачивается в низкочастотной области, наименьшая – в высокочастотной. Поэтому алгоритмы сжатия с потерями стараются избегать внедрения ЦВЗ в высокочастотную область изображения, поскольку встроенный водяной знак, скорее всего, будет уничтожен сжатием. Встраивать ЦВЗ необходимо в среднечастотные и низкочастотные области преобразования контейнера. Сложность возникает при внедрении ЦВЗ в низкочастотную область, содержащую большую часть энергии изображения, потому что неоптимальное внедрение может привести к значительному искажению контейнера. В качестве преобразований могут быть использованы: дискретное косинусное преобразование (ДКП), дискретное вейвлет-преобразование (ДВП), дискретное преобразование Фурье (ДПФ) и так далее.

Рассмотрим один из методов, относящихся к этой группе, метод относительной замены величин коэффициентов ДКП (метод Коха и Жао). Суть метода: первичное изображение разбивается на блоки пикселей. Дискретное косинусное преобразование (ДКП) применяется к каждому блоку. В результате – образуются матрицы коэффициентов ДКП. Каждый блок скрывает один бит данных. Скрытие начинается со случайного выбора блока. Для скрытия 0 стремятся, чтобы разница абсолютных значений коэффициентов ДКП превышала определенную положительную величину, а для скрытия 1 – разница делается меньше некоторой отрицательной величины. Достоинством данного метода является стойкость к компрессии, а недостатком – визуальное ухудшение качества изображения.

Кратко отметим остальные методы [7]. Статистические методы — методы сокрытия данных, при которых изменяются определенные статистические характеристики изображения, при этом получатель способен отличить видоизмененное изображение от исходного.

Методы искажения — методы сокрытия данных, при которых, в зависимости от секретного сообщения, выполняются последовательные преобразования контейнера. В данном методе важно знать первоначальный вид контейнера. Зная различия между первоначальным контейнером и стеганограммой, можно восстановить исходную последовательность преобразований и извлечь скрытые данные. Следует отметить, что при применении данного метода важно соблюдать правило: распространение набора первоначальных контейнеров осуществляется только через секретные каналы доставки.

Структурные методы — методы сокрытия данных, при которых формируется скрываемый текст, посредством осуществления последовательных модификаций частей изображения. Данный метод позволяет не только модифицировать изображение, в котором будет скрыто послание, но и создавать изображение по секретному сообщению. Структурный метод весьма устойчив против атак.

# Глава 2. Метод обнаружения прозрачных

**водяных знаков**

Данная глава является основной в работе. В ней рассматривается метод для обнаружения потенциальных видимых прозрачных водяных знаков на изображении и его применение. Алгоритм поиска потенциальных водяных знаков на изображении реализован в виде программы в среде Wolfram Mathematica 11.3. Код программы приведен в приложении.

**2.1. Описание метода**

Наибольший интерес представляют фотореалистичные изображения, сложность которых подразумевает наличие множества границ. На таких изображениях человеку будет непросто заметить полупрозрачные авторские знаки и логотипы.

После наложении водяных знаков на исходное изображение можно разделить рабочую область полученного изображения на внутреннюю и внешнюю области. Внешней будем считать ту область, которая не изменяется в результате наложения.

С помощью метода обнаружения прозрачных областей можно выдвинуть гипотезы о наличии или отсутствии видимого водяного знака на изображении. Анализ прозрачности в значительной степени сфокусирован на соединениях сегментированных частей, точках переходов. Основное предположение будет заключаться в том, что внешние цвета результирующего изображения можно приближенно взять в качестве внутренних цветов исходного изображения. Чтобы применить метод обнаружения прозрачных областей, потребуется разделение всего изображения на отдельные объекты (области) и обработка каждого из них.

Для разделения изображения на области используем цветовую сегментацию. В пакете Wolfram Mathematica 11.3 доступны несколько методов сегментации изображения с помощью цветовой кластеризации: GaussianMixture, SpanningTree, NeighborhoodContraction, MeanShift, Spectral, DBSCAN, Agglomerate.

Разработанный алгоритм предполагает ряд требований к потенциальным водяным знакам:

1. Водяной знак является n-связной областью;

2. При работе с данными предполагается, что информация содержит только значения цветовых каналов, альфа-канал отсутствует.

3. Внешние цвета результирующего изображения можно приближенно взять в качестве внутренних цветов исходного изображения

4. Водяной знак должен быть получен наложением на исходное изображение картинок, в каждой из которых одна многосвязная область, одинаковый уровень прозрачности (но не 0% и не 100%) и цвет.

Описание алгоритма:

1. Разделим изображение на области с помощью цветовой сегментации.

2. Производим морфологическое разбиение изображения для удаления шумов и правильного разделения на n-связные области.

3. Выберем одну найденную область. Находим границы области и составим для нее множество граничных точек. Проходим по всем точкам границы области, нумеруя каждую из них.

4. Производим интерполяцию сплайнами границы области. В каждой точке полученной фигуры строим нормаль к касательной, беря на линии перпендикуляра точку слева и точку справа, от точки пересечения перпендикуляра и касательной. Составим два массива, состоящих из всех точек слева и всех точек справа.

Если одно изображение получено из другого путем наложения картинки и значения цветов и альфа-канала последней неизвестны, то расчет цвета каждого канала результирующего пикселя после наложения двух пикселей друг на друга находим по формуле

,

где — значение канала пикселя исходного изображения,

— значение канала пикселя накладываемой прозрачной картинки,

— альфа-канал прозрачной картинки,

 — значение канала пикселя результирующего изображения.

Такой способ преобразования данных позволяет находить результирующее изображение для модели RGB.

Запишем данную формулу в следующем виде:

.

Выражение слева назовем отклонением. Применяя формулу, составим уравнение для каждого канала пикселя на границе. Получаем систему из 3 уравнений с четырьмя неизвестными, где — число точек границы. Применяем к полученной системе уравнений метод наименьших квадратов. В итоге получим значения RGB каналов изображения , а также его альфа-канал.

Область считается прозрачной, если альфа-канал решения находится в заданном диапазоне и среднеквадратическое отклонение от выполнения равенства смешивания цветов не превосходит заданного значения.

5. Обрабатываем остальные области морфологического разбиения. Таким образом, исследуем все участки на прозрачность. Выделяем на изображении те области, которые были определены как потенциальные прозрачные водяные знаки.

Основные этапы алгоритма можно изобразить в виде блок-схемы представленной ниже.

Запуск

Сегментация изображения

Морфологическое разбиение сегментированных частей

Нет

Да

Нет

Да

Прозрачная область?

Выбор очередного сегмента и построение границы области

Выделить область цветом

Не выделять  
область

Есть еще области?

Завершение

**2.2. Применение метода в системе информационного поиска**

Когда прозрачные области на изображении найдены, можно выдвинуть различные гипотезы о том, является ли объект водяным знаком или нет. Например, посчитать площадь найденного объекта, а также площадь прямоугольника, в который вписывается объект. Если результат вычислений не превышает верхней заданной границы и находится выше нижней границы, то с некоторой вероятностью можно выдвинуть предположение, что найденный объект является водяным знаком [8]. Обычно подобные маркировки занимают небольшую часть полотна. Окончательное решение выносится экспертом. С помощью подобных манипуляций ускоряется предварительная обработка изображения. Прозрачные слои при большом количестве данных зачастую бывает сложно заметить, однако разметка возможных водяных знаков помогает эксперту быстрее находить проблемные экземпляры.

Задача поиска потенциальных водяных знаков применяется для соблюдения авторских прав, а также для классификации изображений. Используется в системе информационного поиска в средствах массовой информации, социальных сетях, отслеживании нарушений авторских прав.

Предложенный метод поиска потенциальных видимых водяных знаков на изображении реализован в виде программы в среде Wolfram Mathematica 11.3.

Для примера возьмем изображение, на котором присутствует логотип первого канала (рисунок 6). Этот элемент изображения является водяным знаком. Приведем результат работы алгоритма на каждом этапе.



Рисунок 6. Выбранный пример

1. Для разделения изображения на области используем цветовую сегментацию. В пакете Wolfram Mathematica 11.3 изначально доступны несколько методов. Ниже приведены названия методов и время их работы в секундах для данного примера.

GaussianMixture 10

SpanningTree 43

NeighborhoodContraction 43

MeanShift 43

Spectral 43

DBSCAN 62

Agglomerate 74

В дальнейшем будут показаны результаты работы алгоритма после применения метода Spectral для данного примера (рисунок 7).



Рисунок 7. Цветовая сегментация

1. Для последующей обработки изображения требуется провести кластеризацию сегментированных частей (рисунок 8).

Рисунок 8. Результат кластеризация в обычном и бинаризированном видах.

Для удаления шумов и правильного разделения на n-связные области произведем морфологическое разбиение каждого кластера. Приведем пример результата морфологического разбиения в бинаризированном виде (рисунок 9).

Рисунок 9. Пример морфологического разбиения

1. Полученные области нужно исследовать на наличие прозрачности. Обработаем первый элемент и в дальнейшем оценим уровень его прозрачности. Найдем границы этой области и составим множество точек (рисунок 10).

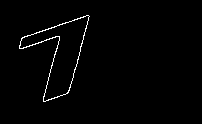


Рисунок 10. Граница области

1. Для более точных вычислений полезно применить интерполяцию границы (рисунок 11), далее будет продемонстрирована интерполяция сплайнами.



Рисунок 11. Пример интерполяции

В каждой точке полученного объекта построим нормаль к касательной, возьмем на линии перпендикуляра точку слева и точку справа от точки пересечения перпендикуляра и касательной. Проходя вдоль всей границы, составим 2 массива, состоящих из всех точек слева и всех точек справа (рисунок 12). В результате в одном массиве будут собраны точки, которые находятся у границы внутри рабочей области, в другом массиве будут точки, находящиеся рядом с границей из внешней области.

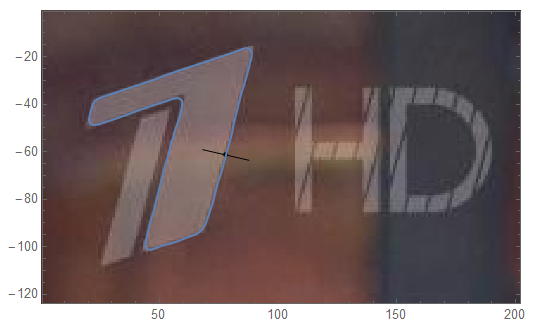


Рисунок 12. Пример перпендикуляра к касательной

Используя массив точек, составим изображение-строку, каждый пиксель которого соответствует цвету точки. Повторим строку 100 раз, чтобы можно было удобно сравнить цвета двух массивов (рисунок 13).





Рисунок 13. Множество точек

Можно заметить, что полученные изображения похожи. Одно из них выглядит более блеклым, чем второе, но между ними есть определенное сходство. Предположим, что одно было получено из другого путем наложения картинки. Значения цветов и альфа-канала последней неизвестны. Расчет цвета каждого канала результирующего пикселя после наложения двух пикселей друг на друга выполняется по формуле

,

где — значение канала пикселя исходного изображения,

— неизвестное значение канала пикселя накладываемой прозрачной картинки,

— альфа-канал прозрачной картинки, который также неизвестен,

 — значение канала пикселя результирующего изображения.

Запишем это выражение в виде

.

Выражение слева назовем отклонением. Применяя формулу, составим уравнение для каждого канала пикселя на границе. Решением будет три значения RGB и альфа-канал. Обозначим неизвестные значения каналов   
 — красный канал,

— зеленый канал,

— синий канал.

Применим к полученной системе уравнений метод наименьших квадратов и найдем , , , . Рассчитаем среднеквадратическое отклонение .

Область считается прозрачной, если альфа-канал решения находится в заданном диапазоне и среднеквадратическое отклонение от выполнения равенства смешивания цветов не превосходит заданного значения. В данном случае значение ~ 0.46, т.е. накладываемое изображение на 46% непрозрачно или на 54% прозрачно. Среднеквадратическое отклонение ~ 3.9. Если задать диапазон , среднеквадратическое отклонение , то область можно считать прозрачной (рисунок 14).



Рисунок 14. Прозрачная область

1. Производим аналогичные действия для следующей области (рисунок 15). Обрабатывая остальные области морфологического разбиения, можно исследовать все потенциальные участки на наличие прозрачности (рисунок 16).

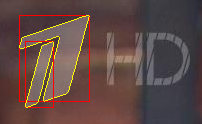


Рисунок 15. Две прозрачные области



Рисунок 16. Все прозрачные области

Экспериментальные результаты показывают, что предложенный метод обнаружения прозрачных областей может обеспечить хорошие результаты при решении задачи поиска потенциальных видимых водяных знаков на изображении. Разметка позволяет эксперту быстро находить и отсеивать проблемные экземпляры, либо отбирать те, с которыми требуется дальнейшая работа. Испытания проводились на фотореалистичных изображениях, на которых присутствовали настоящие полупрозрачные логотипы, что позволяет использовать подобный подход и при работе в реальных условиях. Такой способ предварительной обработки изображений позволяет находить и классифицировать изображения применительно к задаче информационного поиска. Представленный алгоритм можно эффективно применять для данного класса задач.

**Заключение**

В данной работе решены следующие задачи:

1. Проведен анализ поиска прозрачных областей для создания предположений о наличии водяных знаков.
2. Предложен метод обнаружения потенциальных видимых водяных знаков на изображении.
3. Реализован в виде программы в среде Wolfram Mathematica 11.3 представленный алгоритм поиска потенциальных водяных знаков, эффективность которого экспериментально подтверждена.

**Список литературы**

1. Сагайдак Д.А., Файзуллин Р.Т. Способ формирования цифрового водяного знака для физических и электронных документов // Компьютерная оптика. Том 28. 2014. № 1. С.94-104.
2. Красильников Н.Н. Цифровая обработка 2D и 3D изображений: учеб. пособие. – СПб.:БХВ-Петербург, 2011. – 608с.
3. Гришкин В.М., Тимошенко Д.М. Параллельный алгоритм обучения нейронной сети с машиной опорных векторов в качестве выходного слоя // Системы Управления и Информационные Технологии. Том 56. 2014. № 2. С. 75-79.
4. Lin Keming. Research of Digital Watermark Technology Based on Static Image// International Conference on Computer and Communication Technologies in Agriculture Engineering, 2010. <URL:http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5543681>
5. Ahmet Ekin, Radu Jasinschi. Temporal detection and processing of transparent overlays for video indexing and enhancement//Signal Processing Conference, 2005.

URL:<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7077965&tag=1>

1. Manmeet Kaur, Kamna Mahajan. .An Existential Review on Text Watermarking Techniques // International Journal of Computer Applications (0975-8887) Volume 120 – No.1 June 2015. – P. 29-32
2. Конахович Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю. Пузыренко. – Киев: МК-Пресс, 2006. – 288 с.
3. Manish Singh, Xiaolei Huang. Computing Layered Surface Representations: An Algorithm for Detecting and Separating Transparent Overlays Networks// Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2, 2003. URL: <https://pdfs.semanticscholar.org/4abe/cab50c7b1a32057e076c64e74a2564fd757d.pdf>

**Работы автора по теме диссертации**

1\*. Фаустов Б.А. Защита авторских прав в атомной отрасли. Обнаружение и обработка цифровых водяных знаков / Фаустов Б.А., Волошин М.В., Фаустова И.Л. // Изв. Вузов. Физика. – 2017. – Т. 60. – № 11/2. – С. 112-114.

2\*. Фаустов Б.А. Обнаружение цифровых водяных знаков / Фаустов Б.А. – Труды XLIX международной научной конференции аспирантов и студентов «Процессы управления и устойчивость». – Санкт-Петербург. –

2018. (в печати).

**Приложение**

**Программная реализация предложенного алгоритма**

(\*\*\* ЗАГРУЗКА И СЕГМЕНТАЦИЯ ИЗОБРАЖЕНИЯ \*\*\*)

(\* Импорт изображения из файла \*)

img = Import[NotebookDirectory[]<>"in.jpg"];

(\* Получение длины и ширины изображения \*)

imgData=ImageData[img,"Byte"];

width=Length@imgData[[1]];

height=Length@imgData;

Dimensions@imgData

(\* Нахождение границ на изображении \*)

Thinning@EdgeDetect[img,2,Method->"ShenCastan"]

(\* Получение сегментов изображения \*)

data=ClusteringComponents[img,Method->"Spectral",PerformanceGoal->"Quality"];

Dimensions@data

regions=Sort@DeleteDuplicates@Catenate@data

(\* Раскраска сегментов изображения \*)

Colorize@data

(\* Вывод сегментов по отдельности \*)

Table[Image[

Table[

If[data[[i,j]]==n,

Append[imgData[[i,j]],255],

{0,0,0,0}

],

{i,height},{j,width}],

"Byte", ColorSpace->"RGB"],{n,regions}]

(\*\*\* МОРФОЛОГИЧЕСКОЕ РАЗБИЕНИЕ \*\*\*)

(\* пропускать области разбиения по цветам, в которых меньше minCount пикселей \*)

minCount=50;

(\* пропускать области морфологического разбиения,

в которых меньше minCount2 пикселей \*)

minCount2=100;

maxCount2=1/4 height width;

binary=True; (\* получать на выходе бинаризрованные области \*)

borderedImgData=imgData;

morph=Reap[Do[

count=0;

imgData2=Table[

If[data[[i,j]]==n,

count++;

255,

0

],

{i,height},{j,width}];

If[count<minCount,Continue[]];

data2=MorphologicalComponents[Image[imgData2,"Byte", ColorSpace->"Grayscale"]];

regions2=DeleteCases[Sort@DeleteDuplicates@Catenate@data2,0];(\* 0 это фон \*)

Do[

count=0;

startPoints={Infinity,Infinity};

endPoints={-Infinity,-Infinity};

imgData3=Table[

If[binary,

If[data2[[i,j]]==n2,

count++;

startPoints[[1]]=Min[startPoints[[1]],i];

startPoints[[2]]=Min[startPoints[[2]],j];

endPoints[[1]]=Max[endPoints[[1]],i];

endPoints[[2]]=Max[endPoints[[2]],j];

0,

255

],

If[data2[[i,j]]==n2,

count++;

Append[imgData[[i,j]],255],

{0,0,0,0}

]

],

{i,height},{j,width}];

If[count<minCount2||count>maxCount2,Continue[]];

Do[

borderedImgData[[i,startPoints[[2]]]]=borderedImgData[[i,endPoints[[2]]]]={255,0,0},

{i,startPoints[[1]],endPoints[[1]]}];

Do[

borderedImgData[[startPoints[[1]],j]]=borderedImgData[[endPoints[[1]],j]]={255,0,0},

{j,startPoints[[2]],endPoints[[2]]}];

Sow@Image[imgData3,"Byte", ColorSpace->If[binary,"Grayscale","RGB"]],

{n2,regions2}],

{n,regions}]][[2,1]]

(\*\*\* ВЫБОР ИНТЕРЕСУЮЩЕЙ КОМПОНЕНТЫ РАЗБИЕНИЯ \*\*\*)

(\* Максимальная по площади компонента \*)

image=MaximalBy[morph,Count[Catenate@ImageData[#,"Byte"],0]&,1][[1]];

imageData=ImageData[image,"Byte"];

Dimensions@imageData

(\*\*\* ВЫДЕЛЕНИЕ КОМПОНЕНТЫ ПРЯМОУГОЛЬНИКОМ \*\*\*)

startPoints={Min[FirstPosition[#,0,{Length@imageData},{1},Heads->False][[1]]&

/@Transpose@imageData],Min[FirstPosition[#,0,{Length@imageData[[1]]},{1},

Heads->False][[1]]&/@imageData]}

endPoints={Length@imageData-Min[FirstPosition[Reverse@#,0,{Length@imageData},{1},Heads->False][[1]]&

/@Transpose@imageData]+1,Length@imageData[[1]]-Min[FirstPosition[Reverse@#,0,{Length@imageData[[1]]},{1},Heads->False][[1]]&

/@imageData]+1}

Export[NotebookDirectory[]<>"rect.png",

Image[

Table[If[

(i==startPoints[[1]]||i==endPoints[[1]])&&startPoints[[2]]<=j<=endPoints[[2]]||

(j==startPoints[[2]]||j==endPoints[[2]])&&startPoints[[1]]<=i<=endPoints[[1]],

{255,0,0},

imgData[[i,j]]

],{i,height},{j,width}],

"Byte"]

];

(\*\*\* ПОЛУЧЕНИЕ ТОЧЕК ГРАНИЦ \*\*\*)

edgePoints = {#2, -#}&@@@ Position[ImageData[Thinning@EdgeDetect[image,

Method->"ShenCastan"],"Byte"],255,{2}];

ListLinePlot@edgePoints

sqr[x\_]:=x.x

normal[x\_]:={-#2, #}&@@x

(\*\*\* УПОРЯДОЧИВАНИЕ ТОЧЕК ГРАНИЦ \*\*\*)

pointListToLines[pointList\_, neighborhoodSize\_,minLength\_]:= Module[{

NF=Nearest@pointList,

\[Lambda]=Length@pointList,

lineBag = {},

counter = 0,

seenQ,sLB,nextPoint,nearest,couldReverseQ, d,n

},

While[counter < \[Lambda],

sLB = {RandomChoice@DeleteCases[pointList, \_?seenQ]};

seenQ[sLB[[1]]]=True;

counter++;

couldReverseQ=True;

While[

nearest=DeleteCases[NF[Last@sLB, {Infinity, neighborhoodSize}], \_?seenQ];

(nearest=!={} || couldReverseQ),

If[nearest === {},

sLB = Reverse[sLB];

couldReverseQ = False,

nextPoint = MinimalBy[nearest, sqr[Last[sLB]-#]&,1][[1]];

AppendTo[sLB, nextPoint];

seenQ[nextPoint]=True;

counter++;

];

];

AppendTo[lineBag, sLB]

];

Select[lineBag , Length[#] >= minLength&]

]

Table[ListLinePlot[#,AspectRatio->Automatic,PlotRange->{{1,width},-{1,height}}]&

/@pointListToLines[edgePoints, 7,50],{10}]

hLines = pointListToLines[edgePoints, 7,10];

Length@hLines

ListLinePlot[#,AspectRatio->Automatic,PlotRange->{{1,width},-{1,height}}]&/@hLines

(\*\*\* ИНТЕРПОЛЯЦИЯ ГРАНИЦЫ СПЛАЙНАМИ \*\*\*)

(\* Выбор интересующей границы - первой \*)

points=Append[hLines[[1]],hLines[[1,1]]];

fxy=BSplineFunction[points, SplineDegree->30];

(\*\*\* ДЕМОНСТРАЦИЯ НЕПРЕРЫВНОГО ОБХОДА ГРАНИЦЫ \*\*\*)

Manipulate[

Show[

Graphics[{

Inset[img,Scaled[{0.5,0.5}],Automatic,width],

Disk[fxy[p],1],

Line@{

fxy[p]-10Normalize[({#2, -#1}&)@@(D[fxy[s],s]/.s->p)],

fxy[p]+10Normalize[({#2, -#1}&)@@(D[fxy[s],s]/.s->p)]

}

},

AspectRatio->Automatic,

PlotRange->{{1,width},-{1,height}}

],

ParametricPlot[fxy[s],{s,0,1},AspectRatio->Automatic],

ImageSize->800,

Frame->True

]

,{p,0,1}

]

(\*\*\* ПОСТРОЕНИЕ ТАБЛИЦЫ ВНУТРЕННИХ И ВНЕШНИХ ЦВЕТОВ ПО ТОЧКАМ \*\*\*)

count1=count2=0;

colorTab=Table[With[{normal=Normalize[normal@(1/12 points[[p-2]]-2/3 points[[p-1]]+

2/3 points[[p+1]]-1/12 points[[p+2]])]},

With[{p1=Round[points[[p]]-2normal],p2=Round[points[[p]]+2normal]},

If[imageData[[-#2,#1]]&@@p1==0,count1++];

If[imageData[[-#2,#1]]&@@p2==0,count2++];

{

imgData[[-#2,#1]]&@@p1,

imgData[[-#2,#1]]&@@p2

}]

]

,{p,3,Length@points-2}];

{count1,count2}

(\* Первый массив внутренний \*)

If[count1<count2,colorTab=Reverse/@colorTab];

Dimensions@colorTab

Image[ConstantArray[colorTab[[All,1]],100], "Byte"]

Image[ConstantArray[colorTab[[All,2]],100], "Byte"]

(\*\*\* НАХОЖДЕНИЕ ЦВЕТА И ПРОЗРАЧНОСТИ НАКЛАДЫВАЕМОЙ ОБЛАСТИ \*\*\*)

sol = NMinimize[{Sum[sqr[a ({yr,yg,yb}-colorTab[[i,2]])-(colorTab[[i,1]]-colorTab[[i,2]])],

{i, Length[colorTab]}],0<=yr<=255,0<=yg<=255,0<=yb<=255,0<=a<=1},{yr,yg,yb,a}]

Sqrt[ (sol[[1]]/(3 Length@colorTab))] (\* среднеквадратическое отклонение \*)

(\* Вывод накладываемой области \*)

Export[NotebookDirectory[]<>"watermark.png",Image[Table[

If[imageData[[i,j]]==255,

{0,0,0,0},

Round[{yr,yg,yb,255a}/.sol[[2]]]

],

{i,height},{j,width}],"Byte",ColorSpace->"RGB"]];

(\*\*\* ИТОГОВЫЙ АЛГОРИТМ \*\*\*)

regions=Sort@DeleteDuplicates@Catenate@data

(\* пропускать области разбиения по цветам, в которых меньше minCount пикселей \*)

minCount=50;

(\* пропускать области морфологического разбиения,

в которых меньше minCount2 пикселей \*)

minCount2=50;

maxCount2=1/4 height width;

borderedImgData=borderedImgData2=imgData;

Monitor[Do[

count=0;

imgData2=Table[

If[data[[i,j]]==n,

count++;

255,

0

],

{i,height},{j,width}];

If[count<minCount,Continue[]];

data2=MorphologicalComponents[Image[imgData2,"Byte", ColorSpace->"Grayscale"]];

regions2=DeleteCases[Sort@DeleteDuplicates@Catenate@data2,0];(\* 0 это фон \*)

Do[

count=0;

startPoints={Infinity,Infinity};

endPoints={-Infinity,-Infinity};

imageData=Table[

If[data2[[i,j]]==n2,

count++;

startPoints[[1]]=Min[startPoints[[1]],i];

startPoints[[2]]=Min[startPoints[[2]],j];

endPoints[[1]]=Max[endPoints[[1]],i];

endPoints[[2]]=Max[endPoints[[2]],j];

0,

255

],{i,height},{j,width}];

If[count<minCount2||count>maxCount2,Continue[]];

Do[

borderedImgData[[i,startPoints[[2]]]]=borderedImgData[[i,endPoints[[2]]]]={255,0,0},

{i,startPoints[[1]],endPoints[[1]]}];

Do[

borderedImgData[[startPoints[[1]],j]]=borderedImgData[[endPoints[[1]],j]]={255,0,0},

{j,startPoints[[2]],endPoints[[2]]}];

image=Image[imageData,"Byte", ColorSpace->"Grayscale"];

edgePoints = {#2, -#}&@@@ Position[ImageData[Thinning@EdgeDetect[image,

Method->"ShenCastan"],"Byte"],255,{2}];

hLines = pointListToLines[edgePoints, 7,50];

colorTab=Catenate@Table[

points=Join[hLines[[i,-2;;]],hLines[[i]],hLines[[i,;;2]]];

count1=count2=0;

colorTab=Table[With[{normal=Normalize[normal@(1/12 points[[p-2]]-2/3 points[[p-1]]+

2/3 points[[p+1]]-1/12 points[[p+2]])]},

With[{p1={Clip[-#2,{1,height}],Clip[#,{1,width}]}&@@Round[points[[p]]-2normal],

p2={Clip[-#2,{1,height}],Clip[#,{1,width}]}&@@Round[points[[p]]+2normal]},

If[imageData[[#,#2]]&@@p1==0,count1++];

If[imageData[[#,#2]]&@@p2==0,count2++];

{

imgData[[#,#2]]&@@p1,

imgData[[#,#2]]&@@p2

}]

],{p,3,Length@points-2}];

If[count1<count2,colorTab=Reverse/@colorTab];

colorTab,

{i,Length@hLines}];

sol = NMinimize[{Sum[sqr[a ({yr,yg,yb}-colorTab[[i,2]])-(colorTab[[i,1]]-colorTab[[i,2]])],

{i, Length[colorTab]}],0<=yr<=255,0<=yg<=255,0<=yb<=255,0<=a<=1},{yr,yg,yb,a}];

If[

!(0.1<=(a/.sol[[2]])<=0.9)||

sol[[1]]>(0.3 255)^2 3 Length@colorTab,

Continue[]

];

Do[

borderedImgData2[[i,startPoints[[2]]]]=borderedImgData2[[i,endPoints[[2]]]]={255,0,0},

{i,startPoints[[1]],endPoints[[1]]}];

Do[

borderedImgData2[[startPoints[[1]],j]]=borderedImgData2[[endPoints[[1]],j]]={255,0,0},

{j,startPoints[[2]],endPoints[[2]]}];

,{n2,regions2}] (\* цикл по переменной n2 из списка regions2 \*)

,{n,regions}],

Refresh[Column@{ProgressIndicator[n,{1,Length@regions}],

ProgressIndicator[n2,{1,Length@regions2}]},UpdateInterval->1,TrackedSymbols->{}]

];

Export[NotebookDirectory[]<>"out.png",Image[borderedImgData,"Byte"]];

Export[NotebookDirectory[]<>"out2.png",Image[borderedImgData2,"Byte"]];