Минобрнауки России

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт

(технический университет)»

|  |  |  |  |
| --- | --- | --- | --- |
| **Направление подготовки** | | 09.03.01 Информатика и вычислительная техника | |
| **Направленность** | | Систем автоматизированного проектирования и управления | |
| **Факультет** | | Информационных технологий и управления | |
| **Кафедра** | | Систем автоматизированного проектирования и управления | |
| **Курс** | 1 | **Группа** | 404 |

**КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»**

|  |  |
| --- | --- |
| **Тема:** | Разработка приложения для интегрирования водяного знака в изображение |

|  |  |
| --- | --- |
| Выполнил обучающийся | Д. К. Азаров |
| Заведующий кафедрой, проф. | Т. Б. Чистякова |
| Руководитель, доц. | И. Г. Корниенко |
| Консультант, ст. преп. | А. К. Федин |

Оглавление

[1 Введение 3](#_Toc73188483)

[2 Аналитический обзор 4](#_Toc73188484)

[2.1 Обзор и анализ процесса интегрирования водяного знака в изображение. Сравнительная характеристика существующих систем интегрирования водяного знака в изображение. Обоснование актуальности интегрирования водяного знака в изображение. 4](#_Toc73188485)

[2.2 Общая характеристика и особенности интегрирования водяного знака в изображение 6](#_Toc73188486)

[2.3 Обзор и обоснование выбора инструментальных средств разработки приложения для интегрирования водяного знака в изображение 8](#_Toc73188487)

[3 Цель и задачи курсового проекта 10](#_Toc73188488)

[4 Технологическая часть. 11](#_Toc73188489)

[4.1 Формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации. 11](#_Toc73188490)

[4.2 Постановка задачи обработки информации 11](#_Toc73188491)

[4.3 Разработка функциональной структуры приложения для интегрирования водяного знака в изображение 12](#_Toc73188492)

[4.4 Разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение 12](#_Toc73188493)

[4.5 Разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение 13](#_Toc73188494)

[4.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях) 14](#_Toc73188495)

[4.7 Описание структуры программы (модули, основные функции, классы и т. д.) 15](#_Toc73188496)

[4.8 Тестирование программного комплекса (на примере интегрирования водяного знака в изображение) 16](#_Toc73188497)

[5 Итоги выполнения курсового проекта 22](#_Toc73188498)

[6 Список использованной литературы 23](#_Toc73188499)

# 1 Введение

В наше время множество людей выкладывают различные изображения в интернет. Они используются в разных сферах, начиная от ведения блога и заканчивая профессиональным заработком на творчестве. Однако, выкладывая личную работу в интернет, нельзя не обезопасить себя и свои авторские права, поскольку без этого в большинстве случаев изображение попросту украдут для другого сайта или продадут от своего имени. Специально для выхода из такой ситуации есть водяные знаки.

Водяной знак – это отличный способ защитить контент и заявить об авторе изображения. Водяной знак может быть бледным и занимать большую часть изображения, а может разместиться в углу изображения и быть почти незаметным. Здесь всё зависит от цели, с которой вы наносите на изображение водяной знак.

Расположение водяного знака зависит от цели, которую вы преследуете, нанося знак. Чтобы просто указать авторство или брендировать контент, достаточно в одном из углов изображения разместить полупрозрачный знак, который не будет отвлекать на себя внимание. А для защиты изображения лучше разместить знак так, чтобы его было сложно убрать, но вместе с тем он не мешал восприятию изображения. [[1]](#_Список_использованной_литературы)

Водяной знак можно накладывать с помощью множества различных [графический редактор](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80)ов, но обычно они очень сложны и в них надо разбираться. Поэтому существуют приложения, направленные на решение исключительно этой задачи. Самыми популярными программами для наложения водяного знака на изображение являются FastStone Photo Resizer, Easy Image Modifier, BImage Studio. [[2]](#_Список_использованной_литературы)

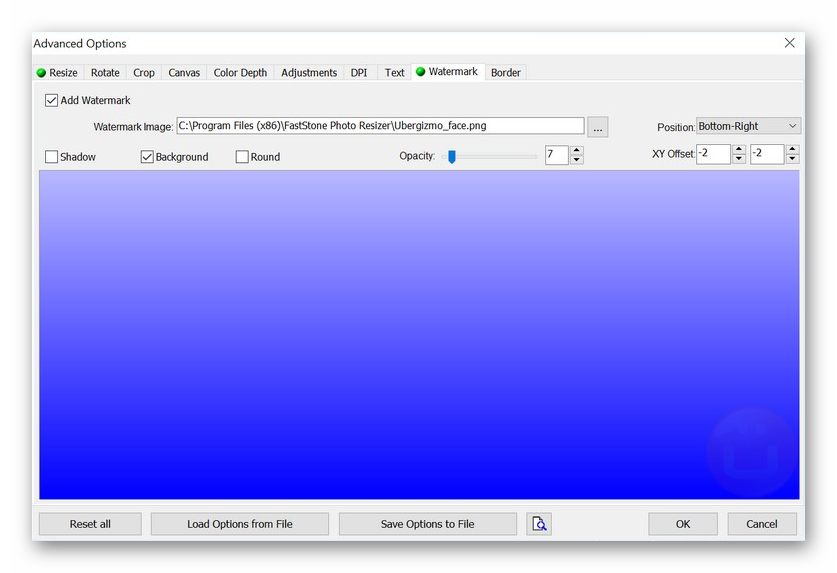
# 2 Аналитический обзор

## 2.1 Обзор и анализ процесса интегрирования водяного знака в изображение. Сравнительная характеристика существующих систем интегрирования водяного знака в изображение. Обоснование актуальности интегрирования водяного знака в изображение.

Программы для интегрирования водяного знака в изображение:

1. FastStone Photo Resizer v4.3

Рисунок 1 – Интерфейс приложения FastStone Photo Resizer



Возможности программы: включение/отключение наложения водяного знака, указание файла, который будет использоваться в этих целях (то есть сам файл у вас должен быть подготовлен заранее), подгон его позиционирования относительно основы, регулировка прозрачности и пара эффектов. Доступна пакетная обработка, что очень актуально в условиях необходимости быстрого нанесения водяного знака на несколько файлов, например, скриншотов для инструкции.

В дополнение разрешается управлять размерами фото, конвертировать их в другие популярные расширения, настраивая параметры процесса преобразования. Помимо всего прочего, в меню расширенных опций кроме раздела с водяным знаком можно найти и другие функциональные вкладки, позволяющие поворачивать, обрезать, отражать файлы, управлять глубиной цветов, добавлять текст, рамки. [[2]](#_Список_использованной_литературы)

1. Easy Image Modifier v4.8

Рисунок 2 – Интерфейс приложения Easy Image Modifier



Чтобы осуществить наложение водяного знака на одно или сразу несколько изображений в этом приложении, надо активировать это средство в разделе добавления водяного знака, напишите текст, который хотите видеть на изображении, укажите параметры шрифта и расположение знака. После этого останется нажать на кнопку «Обработать». В этом приложении нельзя выбрать и наложить какое-то изображение из собственных ресурсов. В связи с этим можно сделать вывод, что софт подходит только для максимально простого способа нанесения водяных знаков, которые часто легко просто убрать, обрезав изображение или отредактировав его в условном Photoshop.

В этом приложении доступны: поворот, отражение, изменение размера, конвертирование в некоторые распространенные форматы. Словом, программа пригодится для максимально простого и незамысловатого редактирования и новичкам, которые не умеют или не хотят создавать персональные водяные знаки, довольствуясь текстовыми средствами защиты, созданными через стандартные шрифты. [[2]](#_Список_использованной_литературы)

1. BImage Studio v1.2.1

Рисунок 3 – Интерфейс приложения BImage Studio



Первый вариант, текстовый, позволяет лишь только ввести слова, изменить цвет и стиль шрифта, указать позиционирование этой надписи. Тут же есть подсказка по возможным вариантам расположения водяного знака. Второй вариант предлагает указать адрес файла с готовым водяным знаком на компьютере, изменить его размер и настроить место, куда он будет помещен.

Из дополнительных возможностей тут есть изменение размера, фильтры (яркость, контраст и пр.), обрезка, поворот и пара других мелких опций. Также здесь имеется пакетная обработка, что поможет быстро нанести водяной знак сразу на все нужные изображения, загрузив их с компьютера и отметив галочками те фото, с которыми будет происходить дальнейшая работа. [[2]](#_Список_использованной_литературы)

В итоге можно сделать выводы, что самыми важными возможностями таких приложений являются: изменение размера, прозрачности и расположения водяного знака.

## 2.2 Общая характеристика и особенности интегрирования водяного знака в изображение

В приложении должны быть реализованы возможности:

**1) Выбор картинок**

у пользователя будет возможность выбрать 2 картинки, которые он хочет преобразовать. Выбранные пользователем картинки будут отображаться.

  
Рисунок 4 – Интерфейс загрузки изображений

**2) Масштабирование**

у пользователя будет возможность изменять размер водяного знака с помощью ползунка. Изменения будут отображаться в отдельном окне.

  
Рисунок 5 – Интерфейс масштабирования

**3) Прозрачность**

у пользователя будет возможность изменять прозрачность водяного знака с помощью ползунка. Изменения будут отображаться в отдельном окне.

  
Рисунок 6 – Интерфейс изменения прозрачности

**4) Место расположения**

у пользователя будет возможность изменять место расположения водяного знака на основном изображении с помощью ползунков. Изменения будут отображаться в отдельном окне.

  
Рисунок 7 – Интерфейс изменения расположения знака

**5) Сохранение изображения**

у пользователя будет возможность выбрать место, в котором он хочет сохранить изображение. Итоговое изображение будет отображаться.

   
Рисунок 8 – Интерфейс сохранения изображения

## 2.3 Обзор и обоснование выбора инструментальных средств разработки приложения для интегрирования водяного знака в изображение

1) Сравнение С++ и Java:

C++ и Java являются языками программирования общего назначения, что означает, что вы можете создавать практически любые типы программных приложений практически для любых платформ, используя правильные инструменты, IDE, библиотеки и фреймворки этих языков.

C++ используется для создания операционных систем, настольных приложений, веб-браузеров, механизмов рендеринга веб-браузеров, библиотек машинного обучения, приложений с тяжелой графической обработкой, баз данных, встроенных систем, мобильных приложений и т.д.

Java используется для создания серверов приложений, веб-приложений, мобильных приложений, настольных приложений, модульных тестов, корпоративных приложений, игр, облачных приложений, веб-API и т.д.

Из списка видно, что С++ больше подходит для поставленной задачи. Также можно учесть, что такая программа как Adobe Photoshop была написана на С++.

Программы на C++ обычно работают быстрее, так как их не нужно интерпретировать, а программы на Java не так быстры, поскольку их нужно сначала интерпретировать, и это требует времени.

Хотя у С++ есть и недостатки перед Java. C++ не является безопасным для памяти языком. Вы можете управлять памятью, что является большим преимуществом. Но могут возникать ошибки памяти и серьезные проблемы, такие как сбои, во время выполнения программы. Java не позволяет манипулировать памятью, она контролируется системой. Так что проблем не возникнет. Java - это язык, безопасный для памяти. [[3]](#_Список_использованной_литературы)

2) Сравнение С++ и С#:

Достоинства С++ были рассмотрены выше, в С# огромным достоинством является возможность легкого создания пользовательского интерфейса, а также функция автоматической сборки мусора. Но несмотря на это большим минусом C# является не кроссплатформенность и то, что компиляция происходит по мере необходимости, то есть во время работы приложения могут быть подтормаживания. [[4]](#_Список_использованной_литературы)

Исходя из перечисленных выше пунктов, наиболее подходящий язык программирования для реализации проекта – C++.

# 3 Цель и задачи курсового проекта

Целью курсового проекта является разработка приложения для интегрирования водяного знака в изображение.

Задача курсового проекта состоит в:

1. написании формализованного описание процесса интегрирования водяного знака в изображение как объекта обработки и информации;
2. постановке задачи обработки информации;
3. разработке функциональной структуры приложения для интегрирования водяного знака в изображение;
4. разработке компонентов математического обеспечения приложения для интегрирования водяного знака в изображение;
5. описании структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях);
6. описании структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях);
7. описании структуры программы (модули, основные функции, классы и т. д.);
8. тестировании программного комплекса (на примере интегрирования водяного знака в изображение);

# 4 Технологическая часть.

## 4.1 Формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации.

На рисунке представлено формализованное описание процесса интегрирования водяного знака в изображение

  
Рисунок 9 – Формализованное описание процесса интегрирования водяного знака в изображение

Где:

X – вектор входных данных;

p – основное изображение;

w – водное изображение;

U – вектор управляющего воздействия;

size – размер;

pos – позиция;

tr – прозрачность;

Y – вектор выходных данных;

r – результирующее изображение;

## 4.2 Постановка задачи обработки информации

Задача состоит в наложении водяного знака на изображение с возможностью изменять размер, место положение и прозрачность знака.

## 4.3 Разработка функциональной структуры приложения для интегрирования водяного знака в изображение

На рисунке представлена функциональная структура приложения для интегрирования водяного знака в изображение  


Рисунок 10 – Функциональной структуры приложения для интегрирования водяного знака в изображение

## 4.4 Разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение

Для наложения водяного знака на изображение используется метод *DrawImage*[[5]](#_Список_использованной_литературы) из класса *Graphics* [[6]](#_Список_использованной_литературы).

Для изменения размера водяного знака разработан метод *changeSizeWatermark* из класса *ImageWork* в этом методе используются формулы:

|  |  |
| --- | --- |
| , | (1) |
| , | (2) |
| , | (3) |
| , | (4) |
| , | (5) |
| , | (6) |
| , | (7) |
| , | (8) |

Для изменения прозрачности знака разработан метод *changeTransparencyWatermark* из класса *ImageWork,* который меняет прозрачность каждого пикселя в градации от 0 до 255.

Для изменения позиции знака разработан метод *changePositionWatermark* из класса *ImageWork.*

Где:

*kw –* отношение ширины основного изображения к ширине водяного знака;

*w1* – ширина основного изображения;

*w2* – ширина водяного знака;

*hn* – новая высота водяного знака;

*kh* – отношение высоты основного изображения к высоте водяного знака;

*h1* – высота основного изображения;

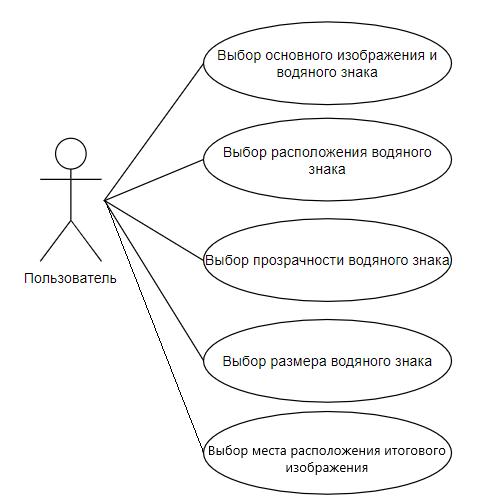
*h2* – высота водяного знака;

*wn* –новая ширина водяного знака;

*s* – новый размер водяного знака, % от основного изображения.

## 4.5 Разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение

Структура пользовательского интерфейса состоит из 3 окон, по которым можно перемещаться между собой, каждое окно предоставляет пользователю свой тип взаимодействия.

  
Рисунок 11 – Структура пользовательского интерфейса

## 4.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях)

Данными в данной программе являются 2 изначальных изображения (основное изображение и водяной знак). Они могу храниться как непосредственно на компьютере, так и на внешнем носителе. Изображения имеют формат \*.png [[7]](#_Список_использованной_литературы), \*.bmp [[8]](#_Список_использованной_литературы), \*.gif [[9]](#_Список_использованной_литературы), \*.jpg [[10]](#_Список_использованной_литературы). Данные форматы были выбраны из-за их популярности. [[11]](#_Список_использованной_литературы)

  
Рисунок 12 – Блок-схема алгоритма интегрирования водяного знака в изображение

## 4.7 Описание структуры программы (модули, основные функции, классы и т. д.)

Таблица 1.1– Класс, составляющий модуль ImageWorkClass.h

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя класса** | **Имя переменной** | **Тип** | **Описание** |
| ImageWork | ­­­­­­­­– | – | Класс для интегрирования водяного знака в изображение |
| \_resultingImage | Bitmap^ | Хранит преобразованное изображение |
| \_mainImage | Bitmap^ | Хранит основное изображение |
| \_mainImagePath | String^ | Хранит путь к основному изображению |
| \_watermark | Bitmap^ | Хранит водяной знак |
| \_watermarkImagePath | String^ | Хранит путь к водяному знаку |
| \_transparencyPixel | array<  bool, 2>^ | Хранит матрицу прозрачности пикселя водяного знака |
| \_alpha | int | Хранит прозрачность изображения |
| \_x | int | Хранит позицию водяного знака по горизонтали |
| \_y | int | Хранит позицию водяного знака по вертикали |
| \_newWidthWatermark | int | Хранит измененный размер водяного знака по горизонтали |
| \_newHeightWatermark | int | Хранит измененный размер водяного знака по вертикали |
| **Метод** | **Тип** | **Описание** |
| changePositionWatermark | Void | Изменяет позицию водяного знака |
| changeTransparencyWatermark | Void | Изменяет прозрачность водяного знака |
| changeSizeWatermark | void | Изменяет размер водяного знака |
| getMainImagePath | String^ | Возвращает путь основного изображения |
| getHeightMainImage | int | Возвращает размер основного изображения по вертикали |
| getWidthMainImage | int | Возвращает размер основного изображения по горизонтали |
| getWatermark | Bitmap^ | Возвращает водяной знак |
| getWatermarkImagePath | String^ | Возвращает путь водяного знака |
| getHeightWatermark | int | Возвращает размер водяного знака по вертикали |

Таблица 1.2– Класс, составляющий модуль ImageWorkClass.h

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя класса** | **Метод** | **Тип** | **Описание** |
|  | getWidthWatermark | int | Возвращает размер водяного знака по горизонтали |
| getResultingImage | Bitmap^ | Возвращает итоговое изображение |

## 4.8 Тестирование программного комплекса (на примере интегрирования водяного знака в изображение)

В ходе тестирования было проверено:

1. Приложение успешно загружает изображения

  
Рисунок 13 – Загрузка изображений

1. При некорректном пути выводится сообщение о ошибке

  
Рисунок 14 – Сообщение о ошибочных данных

1. Водяной знак успешно накладывается на изображение

  
Рисунок 15 – Наложение водяного знака на изображение

  
Рисунок 16 – Наложение водяного знака на изображение

  
Рисунок 17 – Наложение водяного знака на изображение

1. Итоговое изображение успешно сохраняется

  
Рисунок 18 – Сохранение итогового изображения

  
Рисунок 19 – Сохранённое изображение

5)При некорректном пути или названии файла выводится сообщение о ошибке

  
Рисунок 20 – Сообщение о ошибке при некорректном пути

  
Рисунок 21 – Сообщение о существовании файла с данным названием

  
Рисунок 22 – Сообщение при попытке перезаписать изображение, использующееся в программе

# 5 Итоги выполнения курсового проекта

В итоге написания курсового проекта:

1. создан класс для интегрирования водяного знака в изображение;
2. разработан пользовательский интерфейс для интегрирования водяного знака в изображение;
3. написано формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации;
4. поставлена задачи обработки информации;
5. разработана функциональная структура приложения для интегрирования водяного знака в изображение;
6. разработаны компоненты математического обеспечения приложения для интегрирования водяного знака в изображение;
7. описаны структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях);
8. описаны структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях);
9. описаны структуры программы (модули, основные функции, классы и т. д.);
10. протестирован программный комплекс (на примере интегрирования водяного знака в изображение).

# Список использованной литературы

1. Блог Sociate [Электронный ресурс]. – Режим доступа: <https://blog.sociate.ru/brendiruyte-i-zashchishchayte-kontent>/ (Дата обращения 26.05.2021)
2. [lumpics.ru](https://lumpics.ru/) [Электронный ресурс]. – Режим доступа: https://lumpics.ru/software-for-applying-a-watermark-on-a-photo/ (Дата обращения 26.05.2021)
3. ItGap [Электронный ресурс]. – Режим доступа: https://itgap.ru/post/cpp-vs-java (Дата обращения 26.05.2021)
4. C# – Преимущества и недостатки [Электронный ресурс]. – Режим доступа: <https://shwanoff.ru/plus-minus-c-sharp/> (Дата обращения 16.05.2021)
5. Microsoft [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.graphics.drawimage (Дата обращения 16.05.2021)
6. Microsoft [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.drawing.graphics (Дата обращения 26.05.2021)
7. libpng [Электронный ресурс]. – Режим доступа: http://www.libpng.org/pub/png/libpng-1.2.5-manual.html (Дата обращения 26.05.2021)
8. Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/BMP (Дата обращения 26.05.2021)
9. ZX Spectrum [Электронный ресурс]. – Режим доступа: https://zxpress.ru/article.php?id=7463 (Дата обращения 26.05.2021)
10. ZX Spectrum [Электронный ресурс]. – Режим доступа: https://zxpress.ru/article.php?id=7463 (Дата обращения 26.05.2021)
11. Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/JPEG (Дата обращения 16.05.2021)