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 Аналитический обзор 3](#_Toc72955791)

[1.1 Обзор и анализ процесса интегрирования водяного знака в изображение 3](#_Toc72955792)

[1.2 Общая характеристика и особенности интегрирования водяного знака в изображение 6](#_Toc72955793)

[1.3 Обзор и обоснование выбора инструментальных средств разработки приложения для интегрирования водяного знака в изображение 8](#_Toc72955794)

[2 Цель и задачи курсового проекта 10](#_Toc72955795)

[3 Технологическая часть. 11](#_Toc72955796)

[3.1 Формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации. 11](#_Toc72955797)

[3.2 Постановка задачи обработки информации 11](#_Toc72955798)

[3.3 Разработка функциональной структуры приложения для интегрирования водяного знака в изображение 12](#_Toc72955799)

[3.4 Разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение 12](#_Toc72955800)

[3.5 Разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение 14](#_Toc72955801)

[3.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях) 14](#_Toc72955802)

[3.7 Описание структуры программы (модули, основные функции, классы и т. д.) 15](#_Toc72955803)

[3.8 Тестирование программного комплекса (на примере интегрирования водяного знака в изображение) 16](#_Toc72955804)

[4 Итоги выполнения курсового проекта 22](#_Toc72955805)

[5 Список использованной литературы 23](#_Toc72955806)

# 1 Аналитический обзор

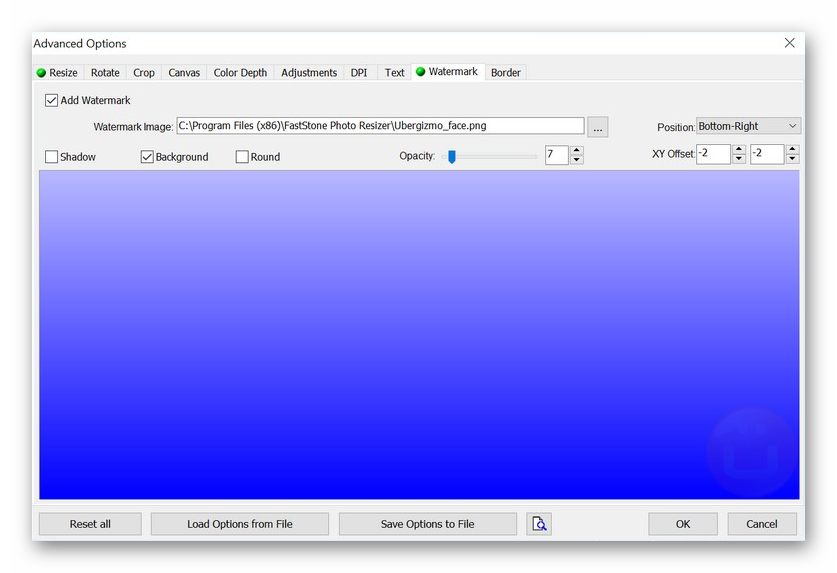
## Обзор и анализ процесса интегрирования водяного знака в изображение

В наше время множество людей создает уникальный контент, среди которого больше всего изображений. 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 различные программы, позволяющие наложить водяной знак.

**Программы для интегрирования водяного знака в изображение:**

1. FastStone Photo Resizer

Рисунок 1 – Интерфейс приложения FastStone Photo Resizer



Параметров у нее не много: 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 вотермарка 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. [[1]](#_Список_использованной_литературы)

1. Easy Image Modifier

Рисунок 2 – Интерфейс приложения Easy Image Modifier



Чтобы осуществить наложение вотермарка на одно или сразу несколько изображений в этом приложении, активируйте 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». Минусом здесь является то, что нельзя в качестве защиты 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.

Других функций здесь не очень-то и много. 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. [[1]](#_Список_использованной_литературы)

1. BImage Studio

Рисунок 3 – Интерфейс приложения BImage Studio



Простая, но в то же время более удобная программа, чем две предыдущих, если дело касается именно наложения водяных знаков. Для выполнения этой задачи софт работает сразу в двух режимах: добавление текста и добавление изображения. Это позволяет пользователю самому выбрать тип вотермарка, отталкиваясь от личных умений в создании данного элемента, однако они оба имеют минимум настраиваемых параметров. 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 вотермарка. Второй вариант предлагает указать адрес файла с готовым водяным знаком на компьютере, изменить его размер и настроить место, куда он будет помещен. На этом функциональность программы в данном плане заканчивается.

Из дополнительных возможностей тут есть изменение размера, фильтры (яркость, контраст и пр.), обрезка, поворот и пара других мелких опций. Словом, BImage Studio — бесплатная, простая и симпатичная программа все для той же быстрой обработки фото, которой сможет пользоваться даже начинающий юзер. Здесь тоже 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. [[1]](#_Список_использованной_литературы)

## 1.2 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

## 1.3 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. [[2]](#_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. [[3]](#_Список_использованной_литературы)

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

# 2 Цель и задачи курсового проекта

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

Задача курсового проекта состоит в:

1) формализованное описание процесса интегрирования водяного знака в изображение как объекта обработки и информации;

2) постановка задачи обработки информации;

3) разработка функциональной структуры приложения для интегрирования водяного знака в изображение;

4) разработка компонентов математического обеспечения приложения для интегрирования водяного знака в изображение;

5) разработка структуры интерфейса пользователя приложения для интегрирования водяного знака в изображение;

6) описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях);

7) описание структуры программы (модули, основные функции, классы и т. д.);

8) тестирование программного комплекса (на примере интегрирования водяного знака в изображение);

# 3 Технологическая часть.

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

  
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;

## 3.2 Постановка задачи обработки информации

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

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

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

Где:

p – основное изображение;

w – водное изображение;

size – размер;

pos – позиция;

tr – прозрачность;

r – итоговое изображение;

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

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

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;

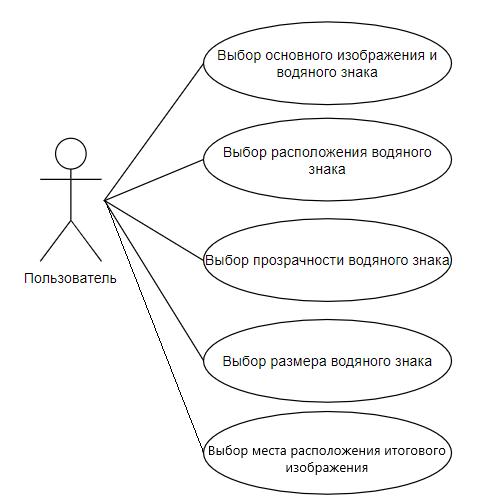
*h2* – высота водяного знака;

*wn* –новая ширина водяного знака;

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

## 3.5 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

## 3.6 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.

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

Таблица 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 |
| getWidthWatermark | int | Возвращает размер водяного знака по горизонтали |
| getResultingImage | Bitmap^ | Возвращает итоговое изображение |

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

В ходе тестирования было проверено:

1. Приложение успешно загружает изображения

  
Рисунок 12 – Загрузка изображений

1. При некорректном пути выводится сообщение о ошибке

  
Рисунок 13 – Сообщение о ошибочных данных

1. Водяной знак успешно накладывается на изображение

  
Рисунок 14 – Наложение водяного знака на изображение

  
Рисунок 15 – Наложение водяного знака на изображение

  
Рисунок 16 – Наложение водяного знака на изображение

1. Итоговое изображение успешно сохраняется

  
Рисунок 17 – Сохранение итогового изображения

  
Рисунок 18 – Сохранённое изображение

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

  
Рисунок 19 – Сообщение о ошибке при некорректном пути

  
Рисунок 20 – Сообщение о существовании файла с данным названием

  
Рисунок 21 – Сообщение при попытке перезаписать изображение, использующееся в программе

# 4 Итоги выполнения курсового проекта

В итоге написания курсового проекта был создан класс для интегрирования водяного знака в изображение. Также был разработан пользовательский интерфейс для интегрирования водяного знака в изображение.

# Список использованной литературы

1. [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)
2. ItGap [Электронный ресурс]. – Режим доступа: https://itgap.ru/post/cpp-vs-java (Дата обращения 26.05.2021)
3. C# – Преимущества и недостатки [Электронный ресурс]. – Режим доступа: <https://shwanoff.ru/plus-minus-c-sharp/> (Дата обращения 16.05.2021)