Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Специальность 230105 Программное обеспечение вычислительной техники и автоматизированных систем
Кафедра автоматизации и компьютерных систем

ДИПЛОМНЫЙ ПРОЕКТ

Тема работы
Приложение для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений

УДК 004.932.1

Студент

Группа	ФИО	Подпись	Дата
З-8001	Шаповалов Александр Викторович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Скирневский И.П.			

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Конотопский В.Ю.	к. э. н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Невский Е.С.			

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АИКС	Фадеев А.С.	к. т. н.		

Томск – 2016 г.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Направление подготовки (специальность) 230105 Программное обеспечение вычислительной техники и автоматизированных систем
Кафедра автоматизации и компьютерных систем

УТВЕРЖДАЮ:
Зав. кафедрой

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Дипломного проекта

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
З-8001	Шаповалову Александру Викторовичу

Тема работы:

Приложение для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений	
Утверждена приказом директора (дата, номер)	№ 2917/с от 15.04.2016

Срок сдачи студентом выполненной работы:	06.06.2016 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Объектом разработки является приложение, выполняющее сегментацию цифровых изображений и оценивает отклонение результатов ручной и автоматической сегментации. Программа должна быть представлена в виде оконного приложения. Исходными данными для разработки дипломного проекта являются следующие документы: – техническое задание на выполнение работ по теме: «Приложение для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений»; – требования к результатам выполнения сегментации цифровых изображений.

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	Разработке подлежат следующие пункты: – реализация пользовательского интерфейса; – реализация алгоритмов сегментации цифровых изображений; – реализация метода оценки отклонения результатов ручной и автоматической сегментации цифровых изображений.
Перечень графического материала *(с точным указанием обязательных чертежей)*	– изображения графического интерфейса; – примеры сегментации изображений; – диаграмма вариантов использования приложения; – диаграммы классов приложения; – диаграммы последовательности объектов приложения.

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Конотопский Владимир Юрьевич
Социальная ответственность	Невский Егор Сергеевич

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Заключение

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	03.03.2016 г.

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Скирневский И.П.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
З-8001	Шаповалов Александр Викторович		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Направление подготовки (специальность) 230105 «Программное обеспечение вычислительной техники и автоматизированных систем»
Уровень образования – специалитет
Кафедра автоматизации и компьютерных систем
Период выполнения – осенний/весенний семестр 2015/2016 учебного года

Форма представления работы:

Дипломный проект

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
20.05.2016 г.	Основная часть	75
24.05.2016 г.	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
24.05.2016 г.	Социальная ответственность	10

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Скирневский И.П.			

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АИКС	Фадеев А.С.	к. т. н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
З-8001	Шаповалову Александру Викторовичу

Институт	электронного обучения	Кафедра	АИКС
Уровень образования	специалитет	Направление/специальность	230105

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала инженерных решений (ИР)*	
2. *Формирование плана и графика разработки и внедрения ИР*	
3. *Обоснование необходимых инвестиций для разработки и внедрения ИР*	
4. *Составление бюджета инженерного проекта (ИП)*	
5. *Оценка ресурсной, финансовой, социальной, бюджетной эффективности ИР и потенциальных рисков*	
Перечень графического материала *(с точным указанием обязательных чертежей)*	
1. *«Портрет» потребителя* 2. *Оценка конкурентоспособности ИР* 3. *Матрица SWOT* 4. *Модель Кано* 5. *ФСА диаграмма* 6. *Оценка перспективности нового продукта* 7. *График разработки и внедрения ИР* 8. *Инвестиционный план. Бюджет ИП* 9. *Основные показатели эффективности ИП* 10. *Риски ИП*	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Конотопский В.Ю.	к. э. н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
З-8001	Шаповалов Александр Викторович		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
З-8001	Шаповалову Александру Викторовичу

Институт	электронного обучения	Кафедра	АИКС
Уровень образования	специалитет	Направление/специальность	230105

Социальная ответственность	Предназначение разделов социальной ответственности
1. Анализ возможных сбоев и их последствий	Сбои при сегментации медицинских изображений и их последствия. Так как приложение может использоваться для сегментации МРТ, рентгеновских снимков и прочего, последствиями ошибки может быть смерть пациента. Сбои при сегментации спутниковых снимков и их последствия. Используя в военных областях может привести с гибели мирного населения. Сбои при сегментации в системах контроля транспортного движения и их последствия. При использовании для наблюдения за нарушениями правил дорожного движения может привести к ошибочным штрафам. Сбой при распознавании отпечатков пальцев и последствия. Область применения расследования происшествий и проверки, при ошибке преступник может оказаться безнаказанным.
2. Безопасное конструирование программного обеспечения	Правильное конструирование обеспечивает эффективность приложения. В данном проекте использовался итеративный подход к разработке. То есть все работы выполнялись с анализом результата и корректированием предыдущих этапов. Данный подход был выбран в связи с тем, что при нем затраты на исправление дефектов обычно ниже, а также различные возможные ошибки выявляются раньше
3. Безопасность пользовательского интерфейса	Пользовательский интерфейс приложения представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с ним. Возможной ошибкой пользователя при работе может быть попытка загрузить в приложение файл неверного формата или же файл, не являющийся изображением, реакцией приложения является сообщение пользователю о том, что выбран файл неверного формата, работа приложения продолжается в обычном режиме и после нажатия на кнопку ОК можно выбрать другой файл. Обеспечена верификация пользовательского ввода и проверка загружаемого файла. Кнопки выполнения сегментации, а также изменения масштаба изображения становятся активными только

	после того как пользователь загрузит в приложение интересующее его изображение.
4. Проблемы смесей гауссовых распределений	При пределе дисперсии стремящемся к нулю логарифм функция правдоподобия стремится к бесконечности, что вызывает крушение кластера. Чтобы избежать подобных особенностей алгоритма было принято решение отслеживать, когда кластер разрушается и сбрасывать его значение до случайной величины в пределах от 0 до 255. Так же обновляется матрица ковариаций данного кластера и затем продолжается оптимизация.
5. Безопасность оборудования	Поскольку разрабатываемый проект представляет собой приложение, то логично, что воздействие факторов, влияющих на физическую платформу, на которой установлено приложение, невозможно предотвратить средствами самого приложения. Для защиты от таких факторов необходимо обеспечить платформу физическими средствами, которые позволят игнорировать данные факторы. Для кратковременного питания компьютера и его устройств при неполадках в сети применяется источник бесперебойного питания (ИБП). Но источник бесперебойного питания подходит только для предотвращения кратковременных неполадок в сети. Для долговременного питания компьютера лучше всего подходит источник резервного питания (ИРП).
6. Технические характеристики	Технические характеристики оборудования, на котором протестирована работоспособность приложения: процессор Intel Core i7-36015QM 2,30 ГГц; ОЗУ 8 ГБ; 64 разрядная система; операционная система Windows 10; видеокарта NVIDIA GeForce GTX 850M. Результаты тестирования так же показали полную совместимость с операционными системами Windows 7 и Windows 8.
7. Требования к персоналу	В связи с опасными последствиями ошибок персонала использующего приложение требуется владеть не только достаточными знаниями в своей области, но и иметь знания в области сегментации цифровых изображений, алгоритмов, а также иметь опыт для выбора подходящих параметров.

Дата выдачи задания для раздела по линейному графику	22.04.2016 г

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Невский Е.С.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
З-8001	Шаповалов Александр Викторович		

РЕФЕРАТ

Выпускная квалификационная работа 85 с., 41 рисунок, 15 таблиц, 22 источника, 1 приложение.

Ключевые слова: ручная сегментация изображений, автоматическая сегментация изображений, к-средних, эффективная сегментация на графах, смеси Гаусса, отклонение.

Объектом исследования является отклонение результатов ручной и автоматической сегментации цифровых изображений.

Цель работы – реализация алгоритмы и оценить отклонения результатов ручной и автоматической сегментации изображений.

В процессе исследования проводился анализ алгоритмов сегментации.

В результате исследования было разработано приложение, выполняющее сегментацию изображений методами: к-средних, смесей Гаусса и эффективной сегментации на графах, а также оценивающее отклонение результатов ручной и автоматической сегментации изображений.

Основные характеристики: высокая точность алгоритма смесей Гаусса, высокая скорость сегментации алгоритма эффективной сегментации на графах.

Область применения: системы контроля дорожного движения, мониторинг, оптическое распознавание символов и т.д.

Экономическая эффективность/значимость работы: данная работа имеет средний уровень научно-технического эффекта, срок окупаемости проекта 1,6 года.

В будущем планируется улучшить реализованные алгоритмы, увеличить количество исследуемых алгоритмов, создать базу изображений с выполненной ручной сегментацией, улучшить алгоритм оценки отклонения.

ОПРЕДЕЛЕНИЯ

В данной работе применены следующие термины с соответствующими определениями:

сегментация изображений: Разделение изображения на регионы или категории, которые соответствуют различным объектам или частям объектов.

алгоритм Краскала: Алгоритм построения минимального остовного дерева взвешенного связного неориентированного графа.

система непересекающихся множеств: Структура данных, которая позволяет администрировать множество элементов, разбитое на непересекающиеся подмножества.

EM-алгоритм: Алгоритм, используемый в математической статистике для нахождения оценок максимального правдоподобия параметров вероятностных моделей, в случае, когда модель зависит от некоторых скрытых переменных.

коэффициент каппа: Показатель точности классификации, использующий результат классификации и проверочный набор данных.

Евклидова норма: Геометрическое расстоянием между двумя точками в многомерном пространстве, вычисляемое по теореме Пифагора.

кластеризация: Задача разбиения множества объектов на группы, называемые кластерами. Внутри каждой группы должны оказаться «похожие» объекты, а объекты разных группы должны быть как можно более отличны.

матрица ковариаций: Матрица, составленная из попарных ковариаций элементов одного или двух случайных векторов.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Сегментацию часто рассматривают как важный шаг в анализе изображений: переход от рассматривания каждого пикселя как единицы измерения к объектам (или частям объектов) в изображении, состоящих из множества пикселей. Хорошо выполненная сегментация облегчает другие стадии анализа изображений.

На сегодняшний день сегментация цифровых изображений применяется в самых разнообразных приложениях, которые включают: оптическое распознавание символов, автоматический осмотр техники, розничная торговля, медицинские изображения, наблюдение.

В связи со сложностью алгоритмов сегментации и разнородностью сфер применения, сегментация часто подвержена ошибкам, поэтому на ряду с автоматическими и полуавтоматическими алгоритмами так же применяется ручная сегментация изображений.

Отклонение результатов ручной и автоматической сегментации цифровых изображений на данный момент является малоизученной областью. Существуют исследования, но в основном применительно к медицинским изображениям, из-за того, что ошибки в данной области наиболее критичны.

Целью данной выпускной квалификационной работы является реализация алгоритма оценки отклонения результатов ручной и автоматической сегментации на основании методов эффективной сегментации изображений на графах [1], к-средних [2] и смесей Гаусса [3].

На основе этих алгоритмов было создано приложение, выполняющее сегментацию цифровых изображений, а также оценивающее отклонение полученных результатов от результатов ручной сегментации. Алгоритм оценки отклонения представляет собой сравнение полученных границ объектов, результат работы которого отображается в процентном соотношении. Для дополнительного контроля отклонения границы можно сравнить визуально-аналитическим методом.

1 АКТУАЛЬНОСТЬ ИССЛЕДОВАНИЯ

Важность и необходимость обработки цифровых изображений проистекает из двух основных области применения: первой является улучшение графической информации для человеческого восприятия, второй обработка данных сцен видео для автоматического машинного восприятия. Цифровая обработка изображений имеет широкий диапазон применения:

– оптическое распознавание символов (чтение рукописных почтовых индексов и автоматическое распознание номерного знака);

– автоматический осмотр (часть инспекции для контроля качества с использованием стереозрения со специальным освещением для измерения допусков на автомобильные корпуса);

– розничная торговля (распознавание объектов для автоматической кассовой полосы);

– медицинские изображения (обнаружение опухолей, диагностика, определение объемов тканей и т.д.);

– наблюдение (мониторинг преступников, анализ трафика дорожного движения, мониторинг бассейнов для поиска тонущих).

На данный момент ни один алгоритм сегментации не дает абсолютно точной информации о границах и областях, вместе с тем что существует огромное количество различных алгоритмов, а также их вариаций. Разработанные системы теперь имеют множество доступных алгоритмов, однако, лишь не многие из них имеют объективные численные оценки.

1.1 Объект исследования

В выпускной квалификационной работе объектом исследования является отклонения результатов ручной и автоматической сегментация цифровых изображений. В приложениях, использующих сегментацию широко применяются такие методы, как: разделение и слияние регионов, эффективная сегментация на графах, водораздел, к-средних, сдвиг среднего, суперпиксели и другие. В связи со сложностью методов было принято решение ограничить

объект исследования тремя методами: к-средние, эффективная сегментация на графах и смеси Гаусса. Выбор первых двух обусловлен популярностью данных методов. Алгоритм сегментации смесями Гаусса не так часто применяется на практике, но является частью более сложного и популярного метода – GrabCut -[4].

При оценке работы алгоритмов в основном сравнивают результаты сегментации друг с другом, либо общая оценка точности выбранного алгоритма.

Jan Mikulka из Технологического Университета Брно исследует ошибки ручной сегментации биомедицинских изображений относительно автоматической сегментации [5]. Но в данной статье не указаны использованные алгоритмы и ошибка рассматривается с точки зрения ошибочного и пропущенного распознавания области.

Доктор Anne Bazille в статье «Impact of Semiautomated versus Manual Image Segmentation Errors on Myocardial Strain Calculation by Magnetic Resonance Tagging» [6] исследует различие результатов ручной, автоматической и полуавтоматической сегментации изображений миокарда полученных с помощью магнитно-резонансной томографии.

В статье Image Segmentation using k-means clustering, EM and Normalized Cuts [7] сравниваются два алгоритма: к-средних и нормализованный разрез графа (Normalized Cut) для сегментации изображений в градации серого. Результаты представлены на рисунке 1 и рисунке 2.

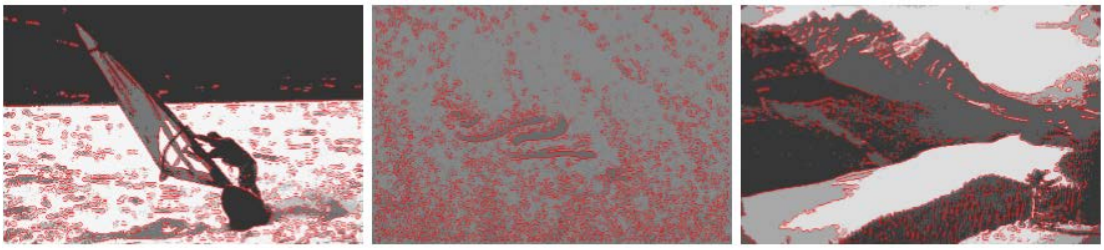

Рисунок 1 – Сегментация методом к-средних

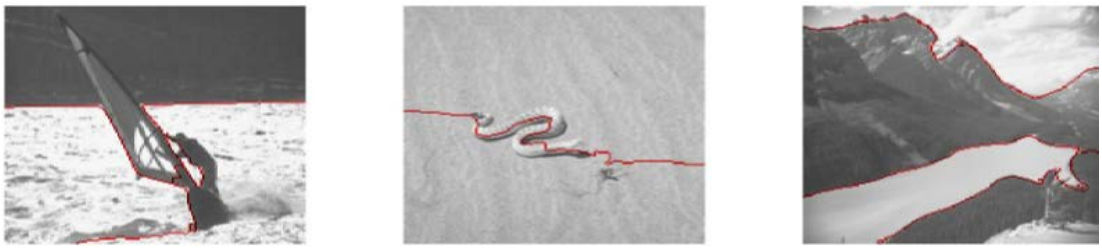

Рисунок 2 – Сегментация методом нормализованный разрез графа

Результатом исследования является заключение, что при небольшом количестве кластеров метод к-средних лучше.

В исследовании K-Means Cluster Analysis for Image Segmentation [8] оцениваются результаты работы алгоритма к-средние (точность, правильность, чувствительность, специфичность сегментации) для четырех изображений, в двух цветовых пространствах: RGB и L*a*b. Изображения, исследованные в данной работе показаны на рисунке 3.

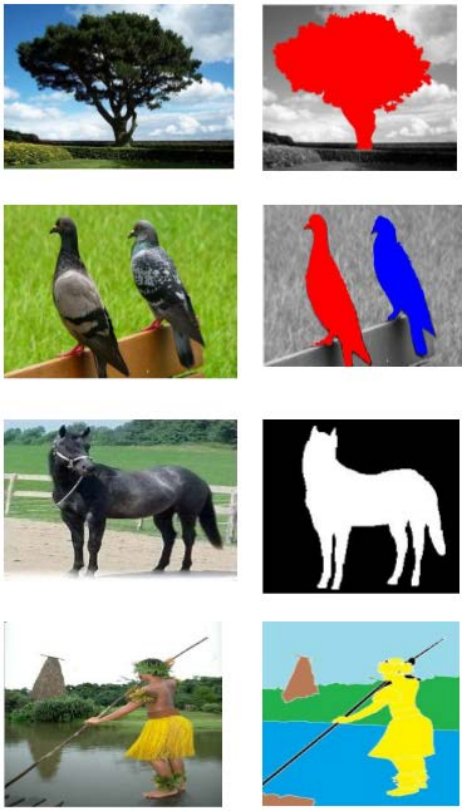

Рисунок 3 – Изображения для анализа сегментации методом к-средних

В результате исследования оказалось, что сегментация изображений в цветовом пространстве L*a*b более точное чем в RGB.

Исследование Caroline Pantofaru [9] в котором сравнивается алгоритм эффективной сегментации на графах, алгоритм сдвиг среднего (mean shift) [10] и гибридный алгоритм в котором сначала используется сдвиг среднего, а затем эффективная сегментация на графах. Гибридный метод создан для того, чтобы сохранить точность алгоритма сдвиг среднего, но вместе с тем уменьшить его зависимость от входных параметров и выбора изображения. Для оценки правильности работы алгоритмов используется случайный индекс нормальной вероятности [11], который облегчает принципы сравнение результата сегментации одинаковых либо разных изображений. Оценивалось точность и стабильность работы алгоритмов. Результатом исследования является заключение, основанное на этих параметрах. В отношении точности сегментации лучшие результаты показал гибридный алгоритм, затем сдвиг среднего и эффективная сегментация на графах. Самым стабильным алгоритмом оказался алгоритм эффективной сегментации на графах.

Существует множество методов оценки результатов сегментации [12]: коэффициент каппа, последовательность сегментации, основанной на регионах [13] и прочие коэффициенты.

В данной работе сравнивается результат автоматической и ручной сегментации. Для этого оценивает пиксельные значения границ обоих изображений. Другими словами пиксели $x_1,\ldots,x_n$ представляющие собой границу объектов на изображении с выполненной ручной сегментацией должны так же являться границами объектов на изображении с выполненной автоматической сегментацией.

2. СЕГМЕНТАЦИЯ ИЗОБРАЖЕНИЙ

Существует три основных подхода к сегментации:

– сегментация, основанная на пороговых значениях (пиксели группируются в зависимости от диапазона значений в которых они лежат);

– сегментация, основанная на регионах (объединяются пиксели, которые являются соседними и имеют похожее значение);

– сегментация, основанная на границах.

2.1 Метод эффективной сегментации на графах

Алгоритм эффективной сегментации на графах был придуман профессором Pedro F. Felzenszwalb и Daniel P. Huttenlocher.

Сегментация изображений, основанная на графах как, правило представляет проблему разделения изображения на регионы в терминах графов:

$$\mathrm{G} = (\mathrm{V}, \mathrm{E}), \qquad (1)$$

где каждый узел $v_i \in V$ соответствует пикселю в изображении;

ребра $(v_i, v_j) \in E$ соединяют некоторые пары соседних пикселей.

Каждое ребро $(v_i, v_j) \in E$ имеет соответствующий вес $\omega((v_i, v_j))$, который представляет собой положительное значение различия (разницы в яркоси, цвете или других локальных свойствах) между соседними пикселями v_i и v_j

Существуют различные способы оценки качества сегментации, но в основном необходимо, чтобы элементы в одной области были похожие. Это означает, что ребра между двумя вершинами в одной компоненте должны иметь небольшой вес, а ребра между вершинами различных компонент должны иметь более высокие веса.

Для реализации метода используется алгоритм Краскала, для более эффективного выполнения алгоритма используется система непересекающихся множеств, которая позволяет быстро объединять множества и узнать какому множеству принадлежит вершина.

На начальном этапе каждый пиксель является вершиной графа и представляет собой собственную область изображения. Затем соединяем ребрами все соседние пиксели, вес ребра определяется Евклидовым расстоянием. В ходе выполнения алгоритма пиксели со схожим цветом объединяются в одну область. Когда сравниваются два пикселя, которые принадлежат разным областям, но имеют схожий цвет, то эти области объединяются в одну.

Качество результата работы алгоритма связано с выбранными входными параметрами. В данном случае это:

– коэффициент для фильтра размытия по Гауссу;
– пороговое значение;
– минимальный размер области.

Пороговое значение и минимальный размер области необходимы для определения границы между разными областями.

Рассмотрим два пикселя в пространстве RGB:

$x_1 = \{R_1, G_1, B_1\}$ и $x_2 = \{R_2, G_2, B_2\}$.

В пространстве RGB цвет определяется значением трех каналов. R – значение красного канала, G – значение зеленого канала, B – значение синего канала. Для того, чтобы узнать схожи ли они по цвету надо определить меньше или больше порогового значения величина веса ребра между ними.

Вес ребра равен Евклидовому расстоянию:

$$\omega = \sqrt{((R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2)}. \quad (2)$$

Существуют объекты, состоящие из частей очень контрастных цветов. Для того чтобы в результате сегментации они были объединены в одну область используется размытие фильтром Гаусса.

Рисунки 4 и 5 отображают пример сегментации изображения методом эффективной сегментации изображений на графах.

Рисунок 4 – Исходное изображение

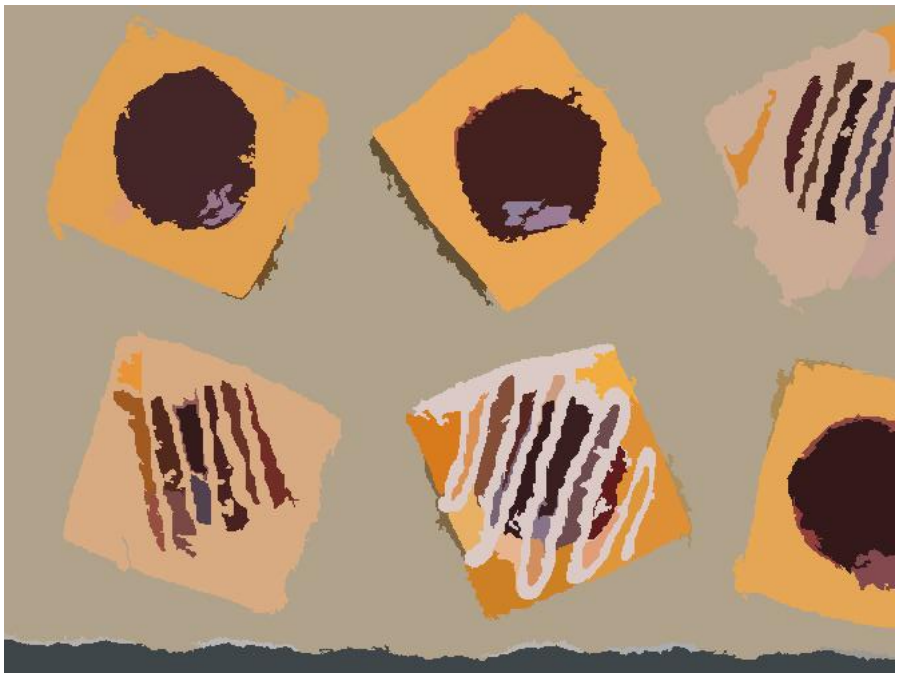

Рисунок 5 – Отсегментированное изображение

Метод эффективной сегментации на графах часто применяется в приложениях и имеет большую скорость сегментации. Пиксели представляют собой вершины графов и на основании Евклидового расстояния и порогового значения группируются, образуя объекты.

2.2 Метод к-средних

Метод к-средних является алгоритмом кластеризации. Рассмотрим проблему идентификации кластеров в многомерном пространстве. Пусть дан набор данных $\{x_1, \dots, x_N\}$ состоящий из N наблюдаемых D-мерных Евклидовых переменных x. Задача состоит в разделении этого набора данных на K кластеров. Определим набор D-мерных векторов μ_k, где $k = 1, \dots K$, в котором μ_k – центр кластера k. Суть алгоритма заключается в минимизации суммарного квадратного отклонения точек от центров кластеров.

Решение этой задачи можно представить в виде двух шагов:

– Присоединение. На основании квадрата Евклидова расстояния присоединяем точки к ближайшему кластеру.

– Обновление. Используя полученные кластеры вычисляем новые центры кластеров на основании среднего значения присоединенных данных.

Пример работы алгоритма в двумерном пространстве изображен на рисунке 6.

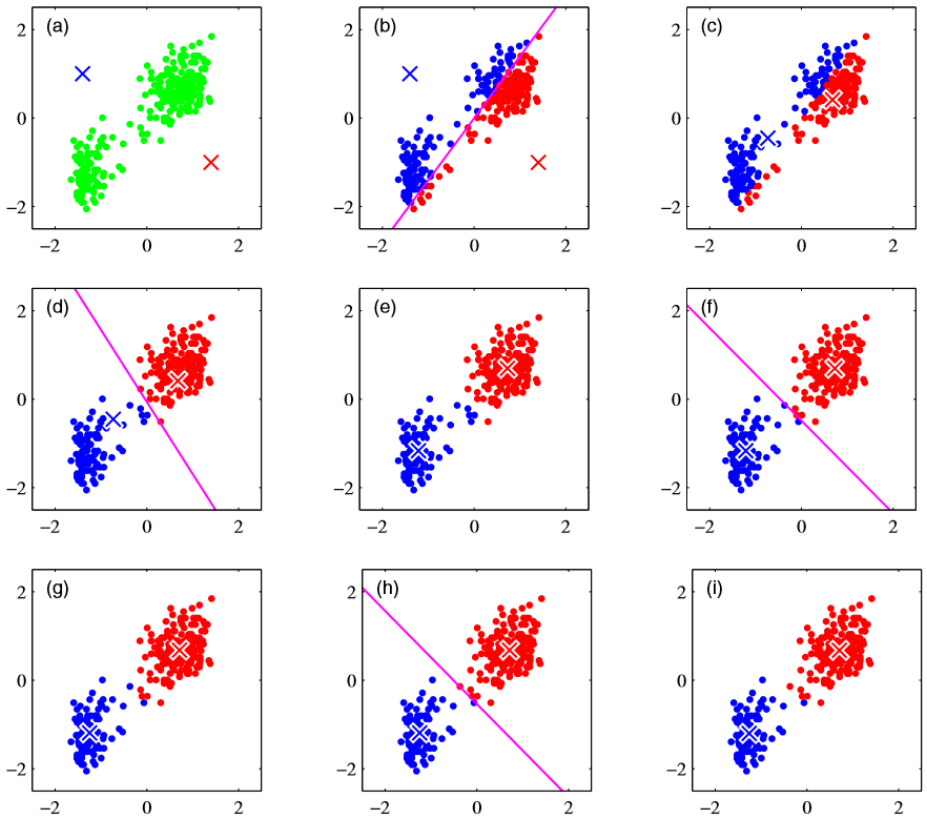

Рисунок 6 – Пример выполнения алгоритма к-средних

Для начальной инициализации центров кластеров применяется метод горной кластеризации [14]. Это простой и эффективный метод, для приблизительной оценки центров кластеров, на основании измерения плотности с помощью горной функции. Метод основан на том, что делает человек визуально разделяя данные на кластеры. Для определения центров кластеров необходимо выполнить три шага:

– сформировать сетку в пространстве данных, где пересечения представляют собой кандидатов в центры кластеров, образующие множество V. Более тонкая сетка увеличивает число потенциальных центров кластеров, но увеличивает требуемые вычисления. Как правило используется равномерное разделение.

– вычисление горной функции, представляющую собой измерение плотности данных. Высота горной функции, для точки v∈V:

$$m(v) = \sum_{i-1}^{N} \exp\left(-\frac{\left||v - x_i|\right|^2}{\alpha^2}\right), \qquad (3)$$

где x_i – точка из набора данных;

α является положительной константой, определяющий область соседних точек.

– выбор центра кластера и последовательное разрушение горной функции. Для этого находим кандидата в центры с максимальной горной функцией. Для вычисления следующего центра кластера необходимо устранить эффект уже идентифицированного кластера, так как чаще всего он окружен большим количеством точек в сетке, которые тоже имеют высокие оценки плотности. Это осуществляется путем пересчета функции горы.

$$m_{new}(v) = m(v) - m(c_1)\exp\left(-\frac{\left||v - c_1|\right|^2}{\beta^2}\right), \qquad (4)$$

где c_1 – идентифицированный центр кластера;

β является положительной константой, определяющий область соседних точек.

В изображении каждый пиксель, а также центр кластера представляет собой точку в 3-мерном пространстве, содержащим интенсивности красного, зеленого и синего каналов:

$$x_1 = \{R_1, G_1, B_1\}, \dots, x_n = \{R_n, G_n, B_n\}, \quad (5)$$

где R – значение красного цвета;

G – значение зеленого цвета;

B – значение синего цвета.

Для каждого пикселя вычисляем расстояние до всех кластеров и присоединяем к тому до которого минимальное расстояние. После прохода всех точек анализируем состав кластеров. Определяем среднее значение цвета {R, G, B} всех пикселей в кластере и полученный цвет назначаем центру кластера. Повторяем до тех пор, пока центры кластеров не перестанут изменяться, либо число итераций не превысит допустимое значение.

После того как алгоритм завершился изменяя значение вектора $\{R, G, B\}$ каждого пикселя значением центра кластера, к которой этот пиксель присоединен обозначаем объекты на изображении.

Фактически задав количество кластеров, мы задаем сколько цветов останется на изображении.

Пример сегментации изображения методом к-средних изображен на рисунках 7 и 8.

Рисунок 7 – Исходное изображение

Рисунок 8 – Отсегментированное изображение

Метод к-средних обычно применяется при небольшом количестве объектов, при большом значении кластеров может служить как алгоритм сжатия. На качество результата влияет начальная инициализация центров кластеров. Для устранения этой проблемы применяется метод горной кластеризации, вычисляющий приблизительные центры. Пиксели присоединяются к ближайшему кластеру и затем на основании этих данных вычисляются новые центры кластеров. Процесс повторяется пока не перестанут изменяться координаты центров кластеров.

2.3 Метод смеси Гаусса

Метод сегментации смесями Гаусса также, как и метод к-средних относится к алгоритмам кластеризации, но в отличии от к-средних вместо вычисления расстояния вычисляется вероятность, и он решает некоторые проблемы идентификации, которые присутствуют в к-средних. В методе к-средних точка может принадлежать только одному кластеру и, если точка находится посередине между двумя кластерами она присоединится только к одному, что фактически является ошибкой. В смесях Гаусса точка может принадлежать нескольким кластерам, а также метод автоматически убирает шумы на изображении.

Рассмотрим d-мерное распределение Гаусса для вектора $x = (x^1, x^2, \dots, x^d)$:

$$N(\mu|x,\Sigma)=\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^{T}\Sigma^{-1}(x-\mu)\right), \quad (6)$$

где μ – центр кластера;

Σ – ковариационная матрица.

Вероятность того, что точка принадлежит к-ой смеси Гаусса определяется функцией правдоподобия:

$$p(x)=\sum_{j}^{k}\omega_j\cdot N(x|\mu_j,\Sigma_j), \quad (7)$$

где ω_j-вес j-ого Гауссиана, $\sum_{j=1}^{k}\omega_j=1$ и $0\leq\omega_j\leq 1$.

Ковариационные матрицы определяют форму кластера.

Существует 3 вида:

– сферическая ковариационная матрица (рисунок 9);
– диагональная ковариационная матрица (рисунок 10);
– полная ковариационная матрица (рисунок 11).

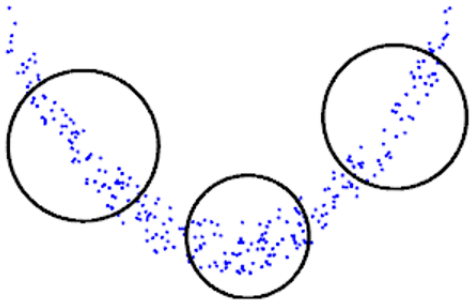

Рисунок 9 – Кластеры при сферической ковариационной матрице

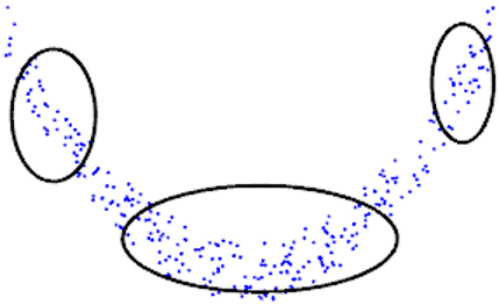

Рисунок 10 – Кластеры при диагональной ковариационной матрице

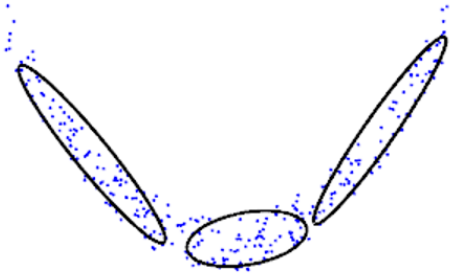

Рисунок 11 – Кластеры при полной ковариационной матрице

Выбор ковариационной матрицы непосредственно влияет на результат. Сферическая имеет наименьшую точность из всех, но она очень быстро рассчитывается, полная ковариационная матрица самая точная, но долго рассчитывается. Диагональная ковариационная матрица, выбранная в данной работе, является наиболее эффективной и имеет хорошую точность.

Цель данного метода максимизировать функцию правдоподобия вместе с параметрами (центры кластеров, матрицы ковариаций и весовые коэффициенты).

Применительно к сегментации изображений получаем трехмерное пространство, тогда на примере одного пикселя и кластера с центром μ и диагональной матрицей ковариаций Σ получаем:

$$x = \begin{matrix} R_x \\ G_x, \\ B_x \end{matrix} \quad (8)$$

$$\mu_n = \begin{matrix} R_\mu \\ G_\mu, \\ B_\mu \end{matrix} \quad (9)$$

$$\Sigma = \begin{matrix} \sigma_R^2 & 0 & 0 \\ 0 & \sigma_G^2 & 0 \\ 0 & 0 & \sigma_B^2 \end{matrix}. \quad (10)$$

Тогда распределение Гаусса, после сокращений будет:

$$\mathrm{N}(\mu|x, \Sigma) = \frac{1}{(2\pi)^{3/2}\sigma_R\sigma_G\sigma_B}\exp\left(-\frac{1}{2\sigma_R^2}\left(R_x\text{-}R_\mu\right)^2 - \frac{1}{2\sigma_R^2}\left(G_x\text{-}G_\mu\right)^2 - \right.$$

$$\left. - \frac{1}{2\sigma_R^2}\left(B_x\text{-}B_\mu\right)^2\right) \quad (11)$$

Один из самых популярных способов максимизировать функцию правдоподобия – использование EM-алгоритма.

Описание алгоритма:

– Инициализация. На этом шаге инициализируем центры кластеров, ковариационные матрицы, весовые коэффициенты и значение логарифма функции правдоподобия. Для лучшей инициализации параметров кластеров используется метод к-средних.

– E шаг. Для каждой точки оцениваем вероятность принадлежности этой точки к каждому кластеру.

– M шаг. Модифицирует параметры, для максимизации функции правдоподобия.

$$\mu_j^{new} = \frac{1}{N_j}\sum_{i=1}^{N} p(x_{ij})x_i, \quad (12)$$

$$\Sigma_j^{new} = \frac{1}{N_j}\sum_{i=1}^{N} p(x_{ij})(x_i - \mu_j)^2, \quad (13)$$

$$\omega_j^{new} = \frac{N_k}{N}, \quad (14)$$

где $N_k = \sum_{i=1}^{N} p(x_{ij})$.

– Вычисляем логарифм функции правдоподобия с новыми параметрами.

Возвращаемся к шагу E до тех пор, пока не выполнится условие:

$$\Delta L = \frac{L_{\text{новое}} - L_{\text{предыдущее}}}{|L_{\text{предыдущее}}|} < 5 * 10^{-4}, \quad (15)$$

где $L = \log \mathrm{p}(X)$.

Алгоритм должен завершиться за 100 итераций.

Пример сегментации изображения методом смесей Гаусса изображен на рисунках 12 и 13.

Рисунок 12 – Исходное изображение

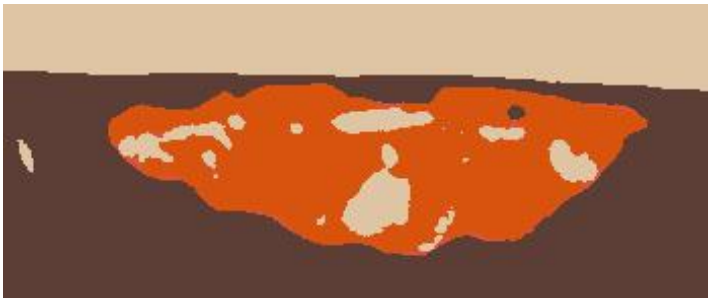

Рисунок 13 – Отсегментированное изображение

Метод смеси Гаусса представляет собой улучшенный и более точный метод к-средних. После инициализации центров кластеров на основании вычисленной функции правдоподобия пиксели присоединяются к необходимому кластеру. Для получения результата необходимо максимизировать данную функцию с помощью EM алгоритма.

2.4 Детектор границ

Целью определения границ объектов является многократное уменьшение количества информации на изображении, сохраняя при этом структурные свойства объектов, которые будут использоваться для дальнейшей обработки изображения. Существует множество алгоритмов выделения границ. В данном проекте применяется детектор границ Canny [15], который разработал John F. Canny в 1986 году. Данный алгоритм оптимален для изображений, на которых наблюдается резкий перепад интенсивности оттенков серого между объектами, что идеально подходит для изображений, к которым применена сегментация.

Алгоритм состоит из пяти шагов:

– размытие изображения;

– поиск градиентов;

– подавление немаксимумов;

– двойная пороговая фильтрация;

– трассировка области неоднозначности.

Размытие изображения необходимо для удаления шумов, но применительно к отсегментированным изображениям данный шаг является лишним.

Алгоритм Canny в основном выделяет края там, где интенсивность серого изменяется в наибольшей степени. Эти области находятся путем определения градиентов изображения. Градиенты в каждом пикселе определяются путем применения оператора Собеля. Для вычисления приближенного значения градиента каждой точки применяются в направлении x и y ядра:

$$G_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}, \qquad (16)$$

$$G_y = \begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}. \qquad (17)$$

Величину градиента можно вычислить, используя Евклидово расстояние:

$$G = \sqrt{\left(G_x^2 + G_y^2\right)} \qquad (18)$$

А также используя Манхэттенское расстояние:

$$G = |G_x| + |G_y| \qquad (19)$$

Используя полученные данные можно вычислить направление градиента:

$$\Theta = \arctan\left(\frac{|G_y|}{|G_x|}\right) \qquad (20)$$

Подавление немаксимумов, двойная пороговая фильтрация и трассировка области неоднозначности необходимы, чтобы границы стали более тонкими и четкими, убирая из границ лишние пиксели.

3 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

Для создания приложения был выбран итеративный подход, в качестве типа приложения было выбрано клиентское приложение [16].

На первой итерации разрабатывалась часть приложения для выполнения эффективной сегментации на графах, на второй итерации реализовывался метод к-средних, на третьей метод смесей Гаусса и на четвертой оценка отклонения результатов ручной и автоматической сегментации изображений.

3.1 Анализ функций системы

В процессе анализа технического задания, приложение совершает следующие действия:

– выбор изображения из диалогового окна;

– выбор изображения путем перетаскивания файла изображения в окно приложения;

– отображение выбранного изображения;

– изменение масштаба отображения;

– выполнение сегментации;

– вывод результата сегментации на экран;

– сохранение результата сегментации в файл;

– анализ отклонения результатов ручной и автоматической сегментации и вывод данных на экран.

Исходя из анализа действий, которые должно осуществлять приложение, была разработана UML-диаграмма вариантов использования (рисунок 14).

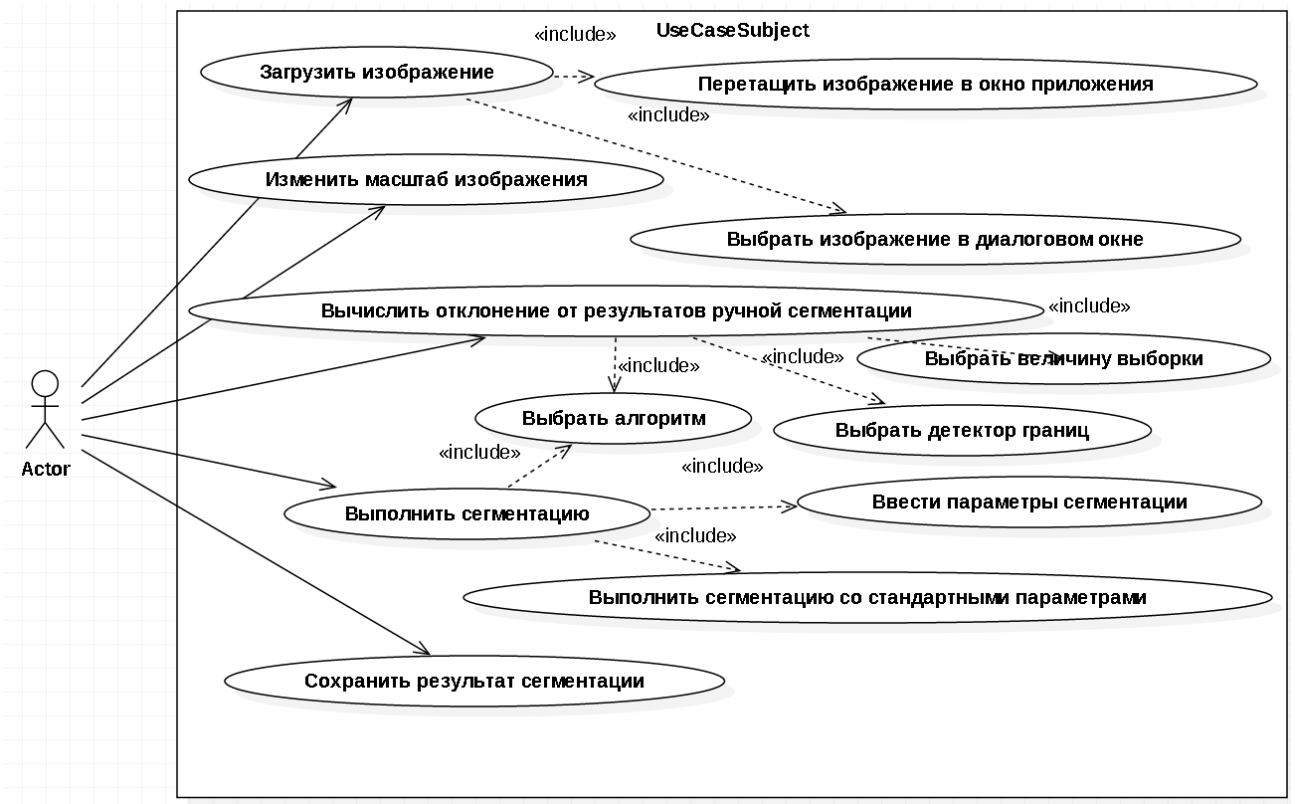

Рисунок 14 – Диаграмма вариантов использования

3.2 Архитектура приложения

На основании анализа теоретических сведений был реализован метод эффективной сегментации на графах, диаграмма классов которого изображена на рисунке 15.

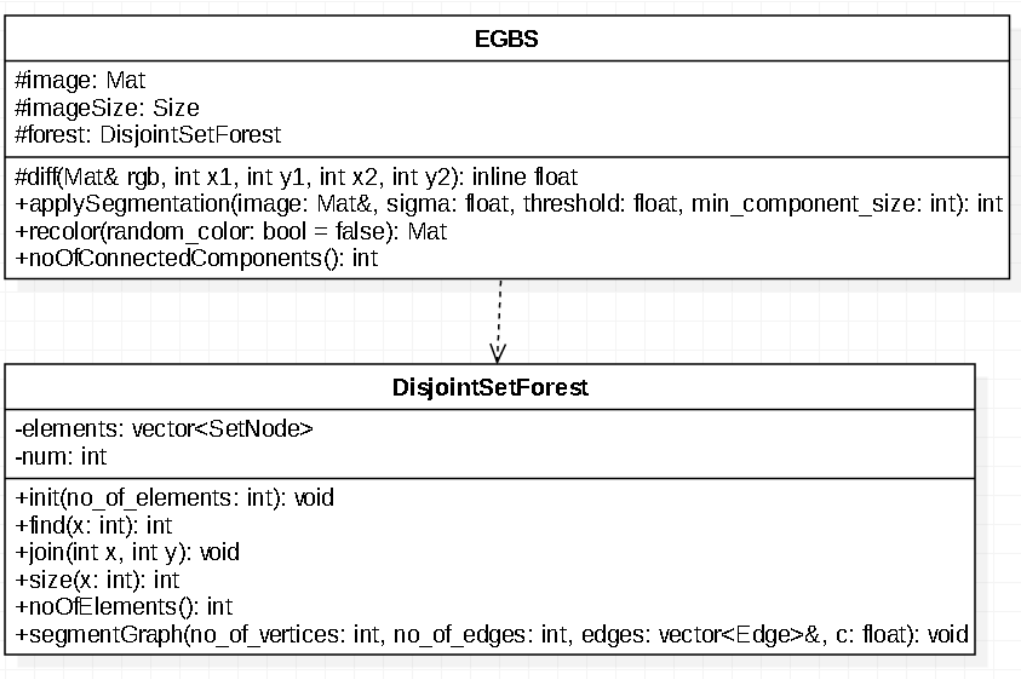

Рисунок 15 – Диаграмма классов метода эффективной сегментации на графах

Реализация метода состоит из двух классов:

– класс DisjointSetForest представляет собой систему непересекающихся множеств;

– класс EGBS является главным классом данного метода и служит для выполнения сегментации.

Листинги классов расположены в Приложении А.

Процесс сегментации изображения можно схематично отобразить посредством диаграммы последовательности, изображенной на рисунке 16.

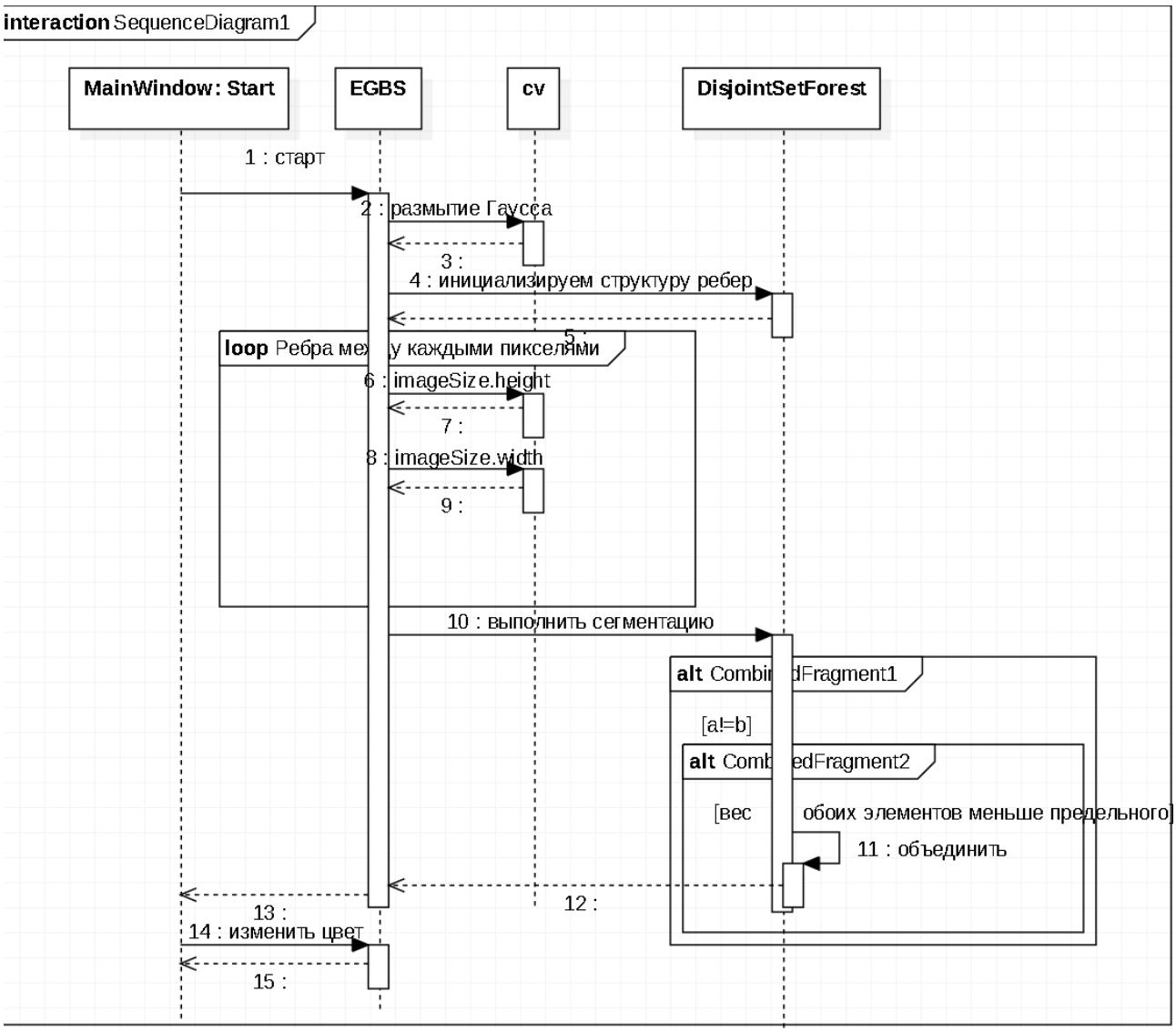

Рисунок 16 – Диаграмма последовательности метода эффективной сегментации на графах

После запуска сегментации производится размытие изображения по Гауссу, алгоритм которого реализован в библиотеке OpenCV, затем инициализируем структуру, представляющую собой систему непересекающихся множеств.

Следующим шагом в цикле создаются ребра с весом между каждыми пикселями. Далее выполняется сегментация (если вес ниже заданного порогового значения элементы объединяются). Заключительным этапом является изменение цвета полученных групп пикселей.

Описанный в теоретической части алгоритм к-средних представлен на диаграмме классов рисунок 17.

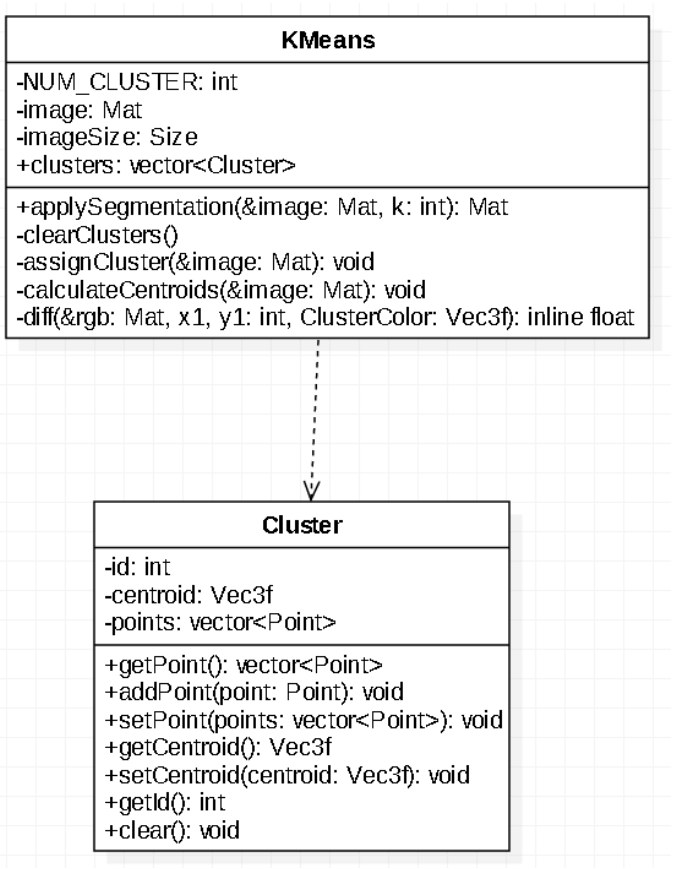

Рисунок 17 – Диаграмма классов метода к-средних

Реализация состоит из двух классов:

– класс Cluster представляет собой кластер для хранения точек расстояние, до которого наиболее минимальное;

– класс KMeans выполняет сегментацию.

Листинг классов расположен в Приложении А.

Процесс сегментации и взаимодействия объектов изображен на диаграмме последовательностей рисунок 18.

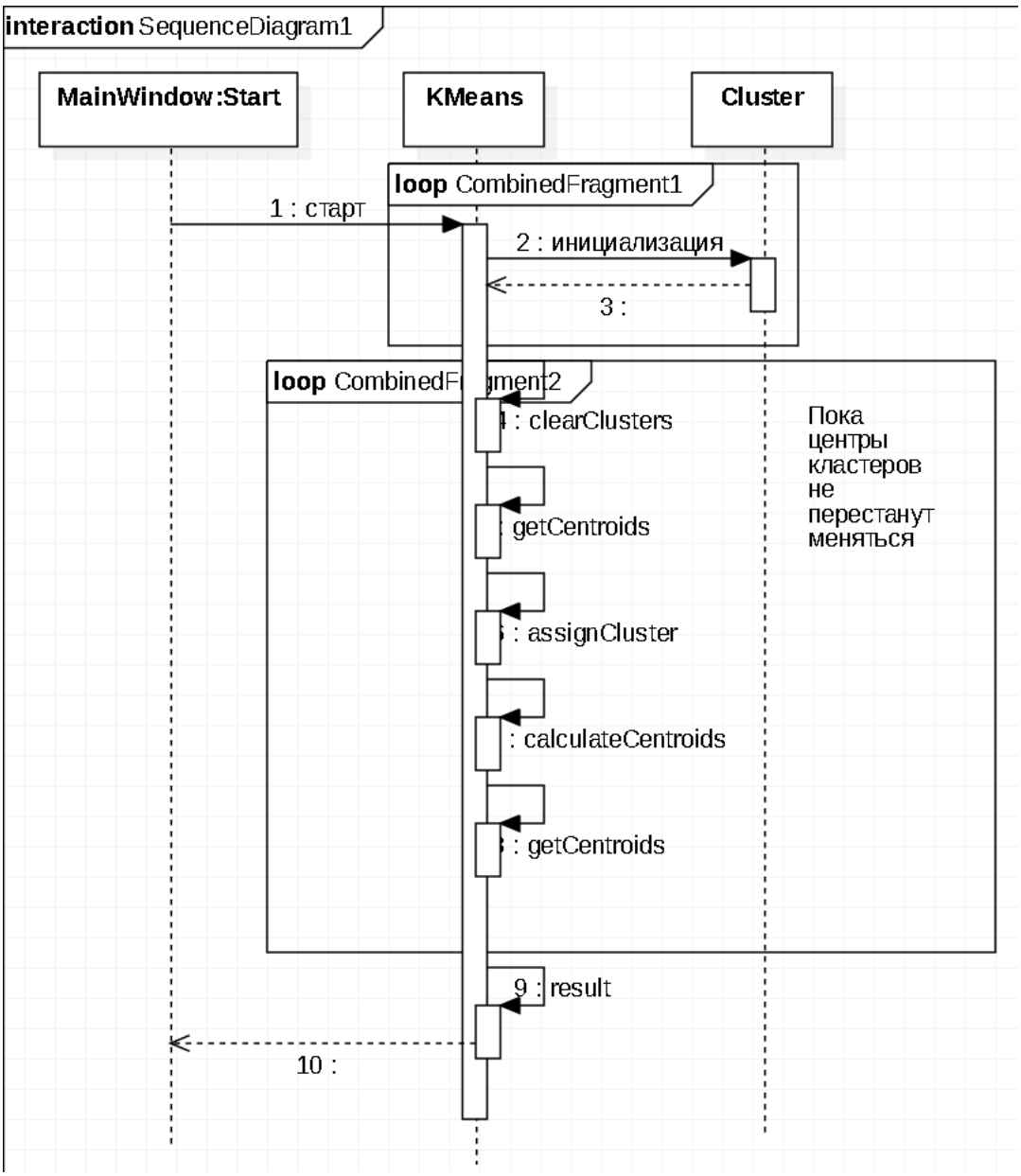

Рисунок 18 – Диаграмма последовательности метода к-средних

Пользователь выбирает количество кластеров, после запуска сегментации методом к-средних инициализируются кластеры и их центры в виде вектора случайных значений, представляющих собой цвет данного кластера. Происходит запуск алгоритма, описанный в теоретической части. Когда центры кластеров перестают изменять свое значение цикл завершается. Завершающим этапом происходит изменение цвета всех точек, присоединенных к кластеру цветом, соответствующим центру кластера, к которому они присоединены.

Анализом теоретических сведений относительно метода смесей Гаусса являются классы, изображенные на рисунке 19.

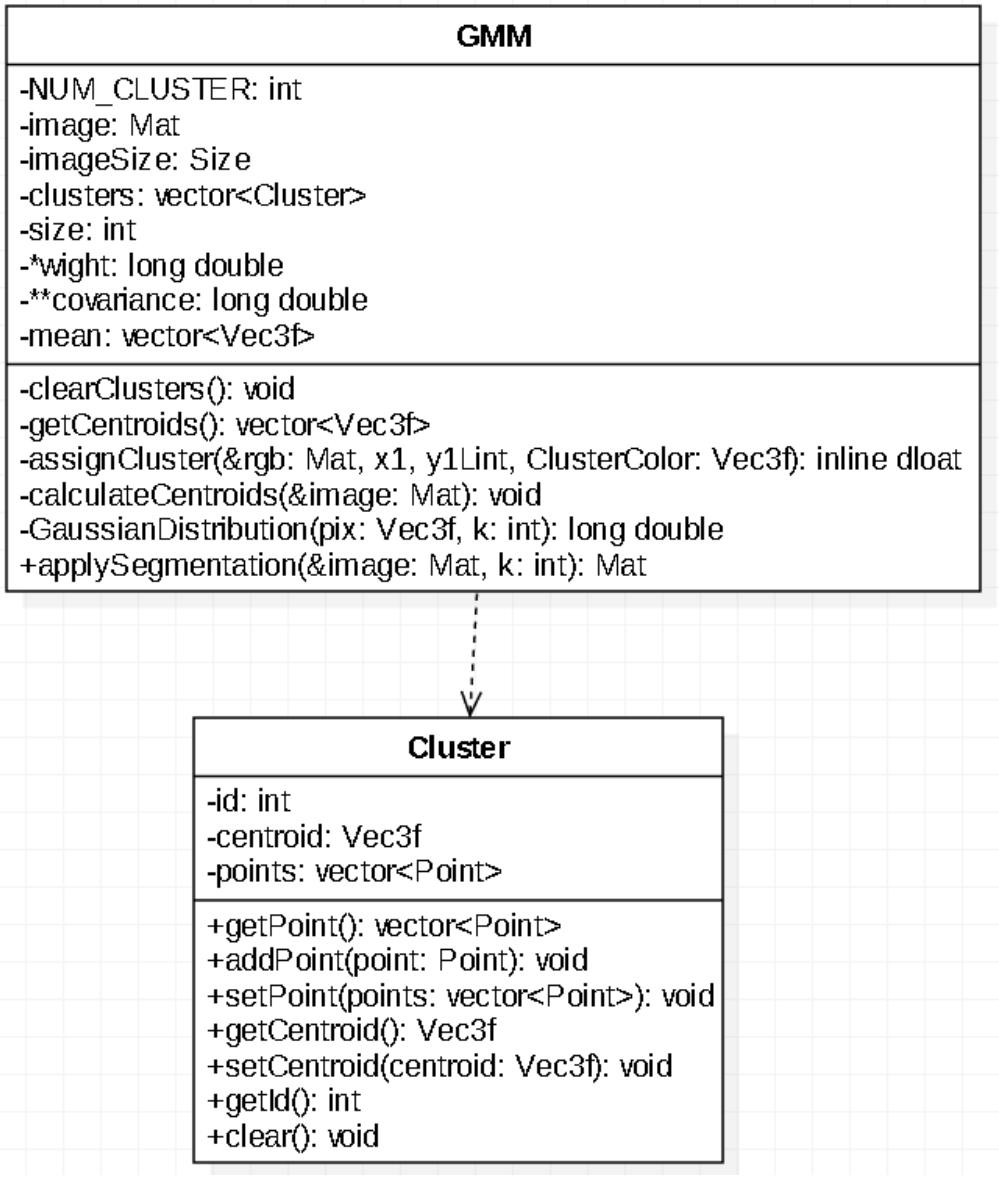

Рисунок 19 – Диаграмма классов метода смесей Гаусса

Реализация метода включает в себя два класса:

– класс Cluster представляет собой кластер для хранения точек расстояние, до которого минимальное;

– класс GMM выполняет сегментацию по средствам EM алгоритма.

Листинги классов расположены в Приложении А.

Процесс сегментации и взаимодействия объектов изображен на диаграмме последовательностей рисунок 20.

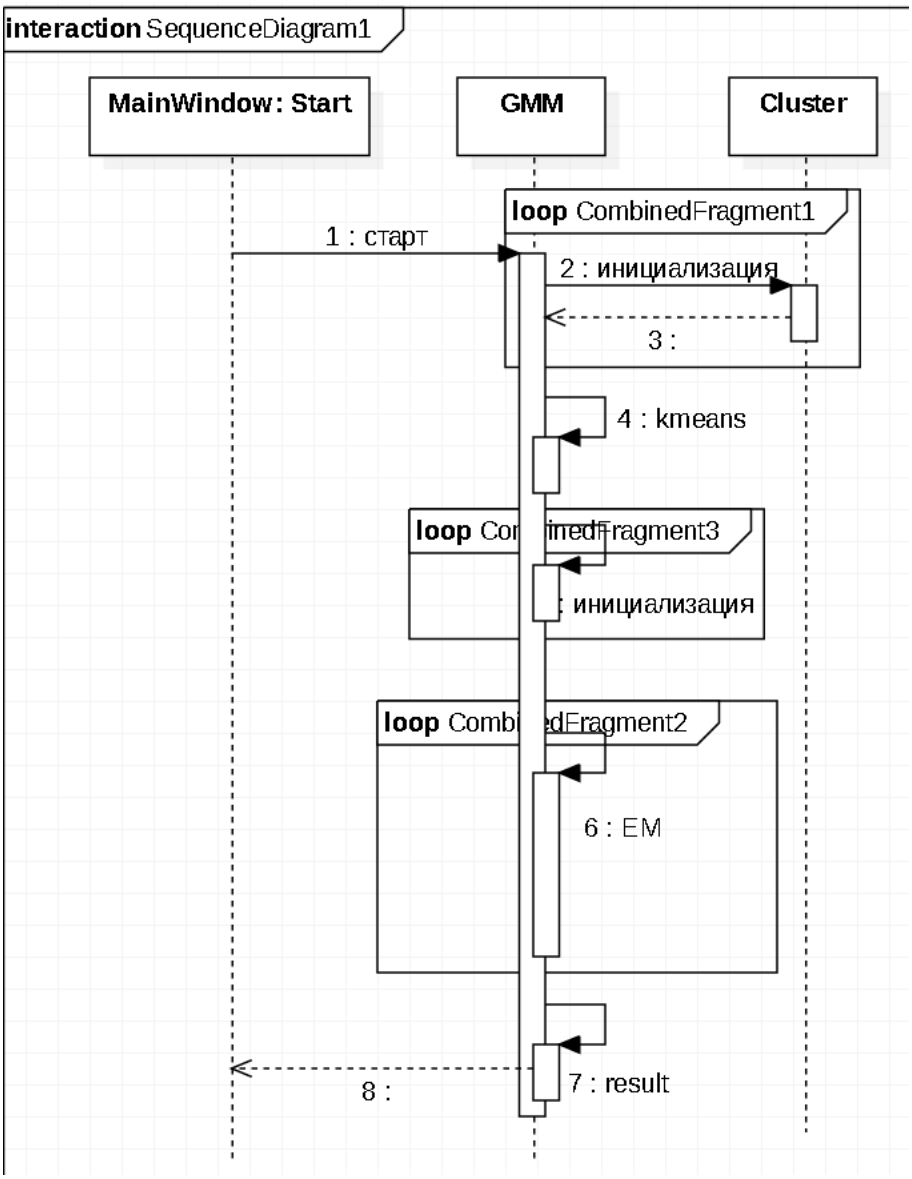

Рисунок 20 – Диаграмма последовательности метода смесей Гаусса

На основании вариантов использования, разработанных методов сегментации и требований было создано приложение для оценки отклонения результатов ручной и автоматической сегментации данных. Диаграмма классов, демонстрирующая работу приложения, взаимодействие классов и связи между ними изображены на рисунке 21.

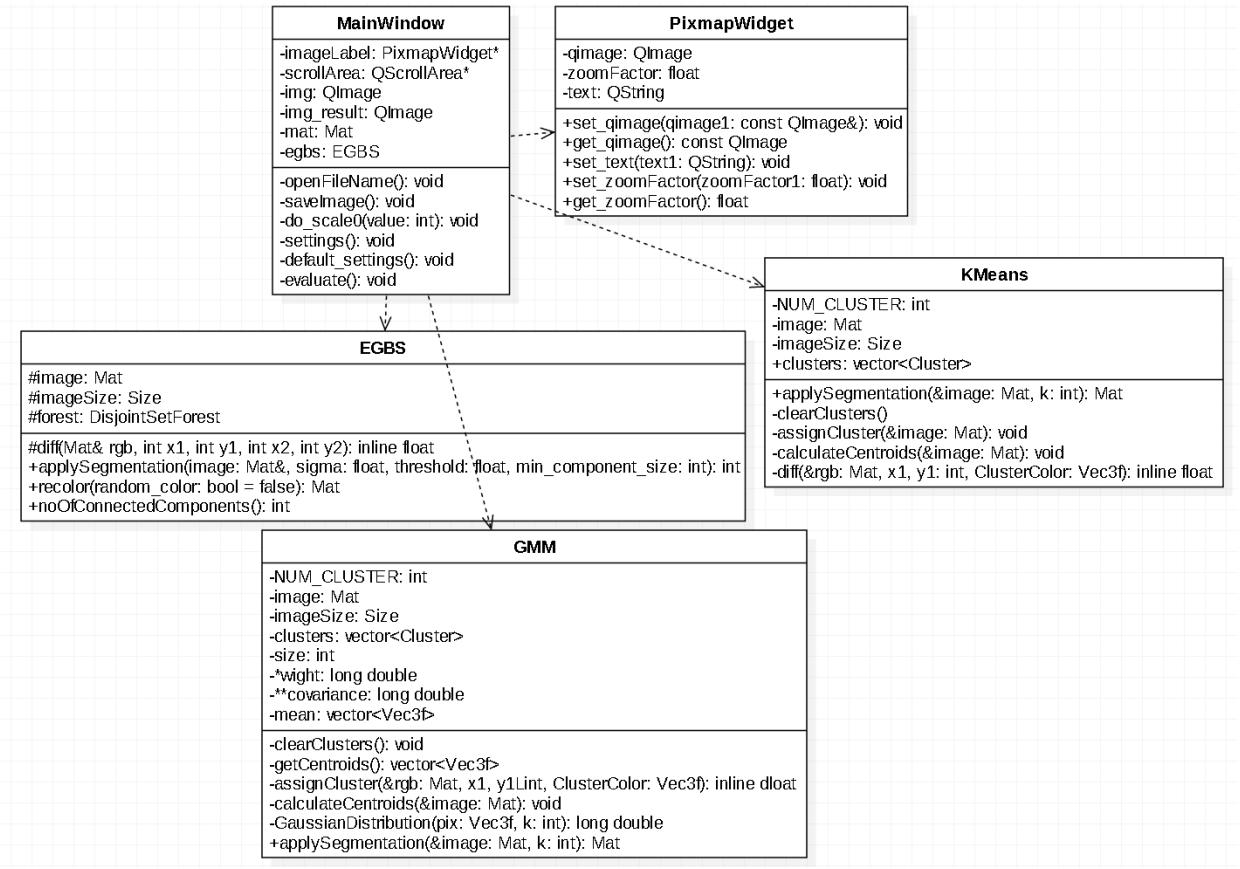

Рисунок 21 – Диаграмма классов приложения для оценки результатов ручной и автоматической сегментации цифровых изображений

Как видно из рисунка 21, класс MainWindow представляет собой главное окно приложения и служит для взаимодействия пользователя с системой, выполнения сегментации и оценки отклонения результатов ручной и автоматической сегментации изображений. Также реализован класс PixmapWidget который служит для хранения, отображения и изменения масштаба изображений.

В приложении реализована функция сохранения настроек. Размер и положение главного окна, выбранный алгоритм и его параметры сохраняются после закрытия приложения.

Листинг данных классов расположен в Приложении А.

Процесс оценки отклонения результатов автоматической и ручной сегментации изображен на рисунке 22.

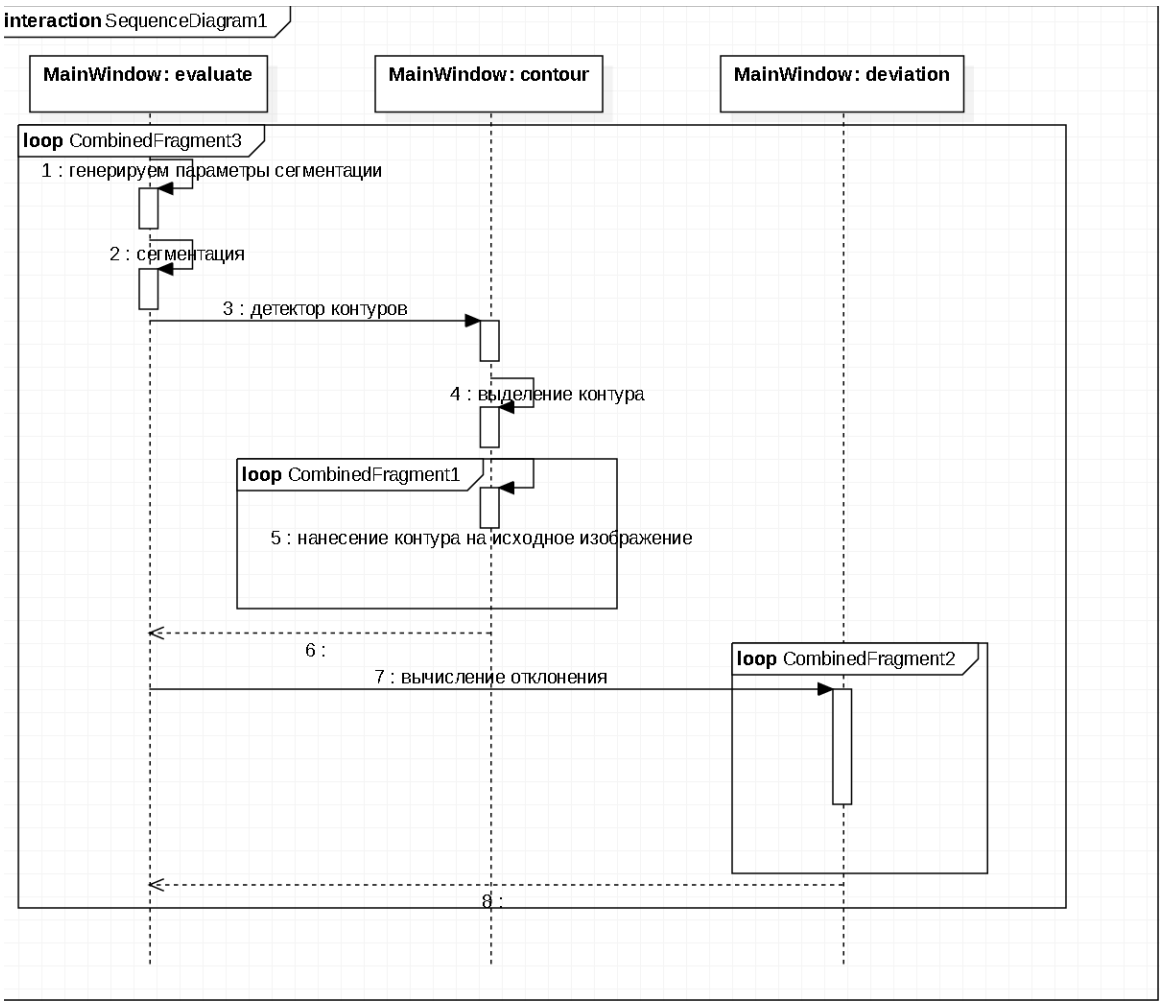

Рисунок 22 – Диаграмма последовательности оценки отклонения

4 РАЗРАБОТКА ПРИЛОЖЕНИЯ

4.1 Выбор инструментов и средств разработки

При выполнении были использованы следующие программные средства:

– интегрированная среда разработки QT Creator 4.0.0 (версия библиотеки QT 5.6.0);

– библиотека OpenCV 2.4.13;

– программный инструмент моделирования StarUML 2.6.0;

– операционная система Windows 10;

– графический редактор GIMP 2.8.

Выбор среды разработки QT [17] обусловлен:

– кроссплатформенностью;

– наличием подробной документации и множества примеров;

– система сигналов и слотов;

– возможность работы с системами контроля версий;

– простое создание интерфейса.

Использование библиотеки OpenCV [18] не являлась обязательным условием выполнения работы, алгоритмы сегментации, реализованные в ней в данном приложении, не использовались. Область применения данной библиотеки является работа с пикселями и конвертирование изображения. Данный выбор сделан на основании того, что классы, реализованные в QT для работы с изображениями (QImage, QRgb) не являются столь эффективными, как в OpenCV.

Выбор языка программирования C++ [19] обусловлен:

– подробная документация;

– поддержка различных стилей программирования;

– программы, написанные на c++ быстро выполняются;

– наличие стандартов c++11 и c++14.

Графический редактор GIMP использовался для выполнения ручной сегментации изображений. Данный графический редактор является бесплатным и имеет мощные инструменты для редактирования изображений.

Приложение StarUML 2.6.0 использовался для проектирования программного обеспечения.

4.2 Демонстрация работы приложения

На основании требований к разрабатываемому приложению и анализа спроектированных методов был разработан интерфейс приложения.

Главное окно приложения изображено на рисунке 23.

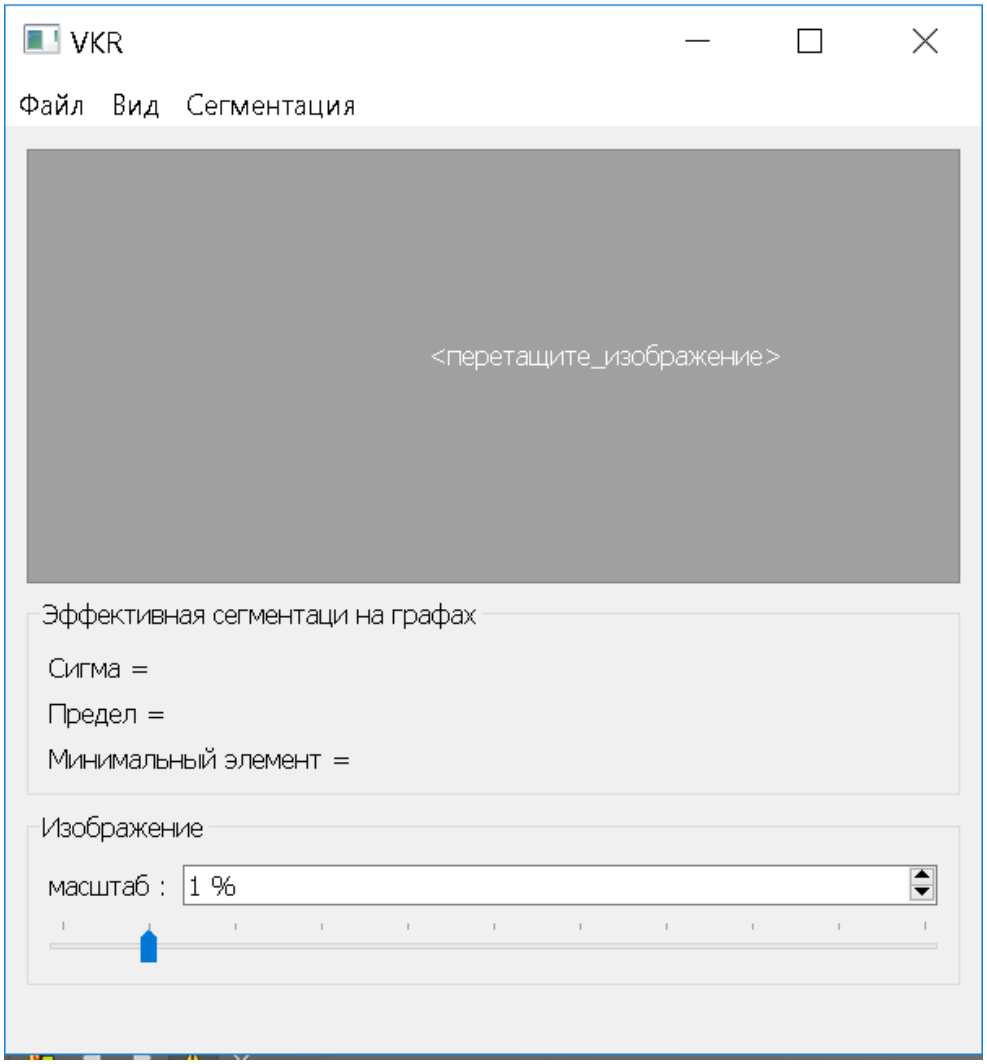

Рисунок 23 –Главное окно приложения

Интерфейс состоит из виджета отображения изображений, группы для отображения параметров выбранного алгоритма сегментации, и группы

отвечающей за изменение масштаба изображения. Приложение содержит три меню:

– файл;
– вид;
– сегментация.

Меню файл содержит основные функции управления приложением. Оно позволяет открыть файл изображения, сохранить отображаемое на виджите изображение (исходное или изображение, к которому применена сегментация) и закрыт ь программу. Также в нем отображается список из 5 последних открытых изображений. Структура изображена на рисунке 24.

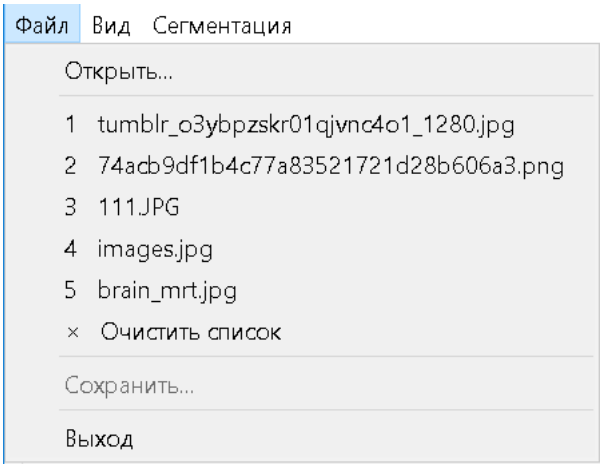

Рисунок 24 – Меню файл

Меню вид служит для управления отображением изображений. Оно позволяет уменьшать, увеличивать изображение, вернуть исходный размер и для функций изменения масштаба имеет горячие клавиши. Внешний вид показан на рисунке 25.

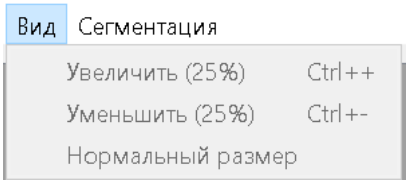

Рисунок 25 – Меню вид

Меню сегментация позволяет запустить выбранный алгоритм сегментации, перейти в окно настроек или в окно оценки отклонения результатов

ручной и автоматической сегментации. Внешний вид меню показан на рисунке 26.

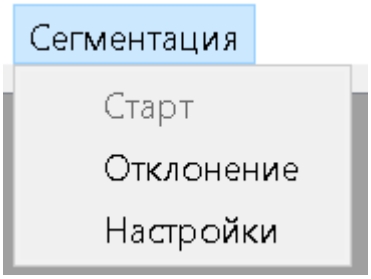

Рисунок 26 – Меню сегментация

Кнопка старт становится активной только после того, как в приложение будет загружено изображение.

Диалоговое окно настроек позволяет выбрать алгоритм сегментации и параметры его запуска. Так же можно сбросить параметры на стандартные с помощью кнопки Reset. Внешний вид она показан на рисунке 27.

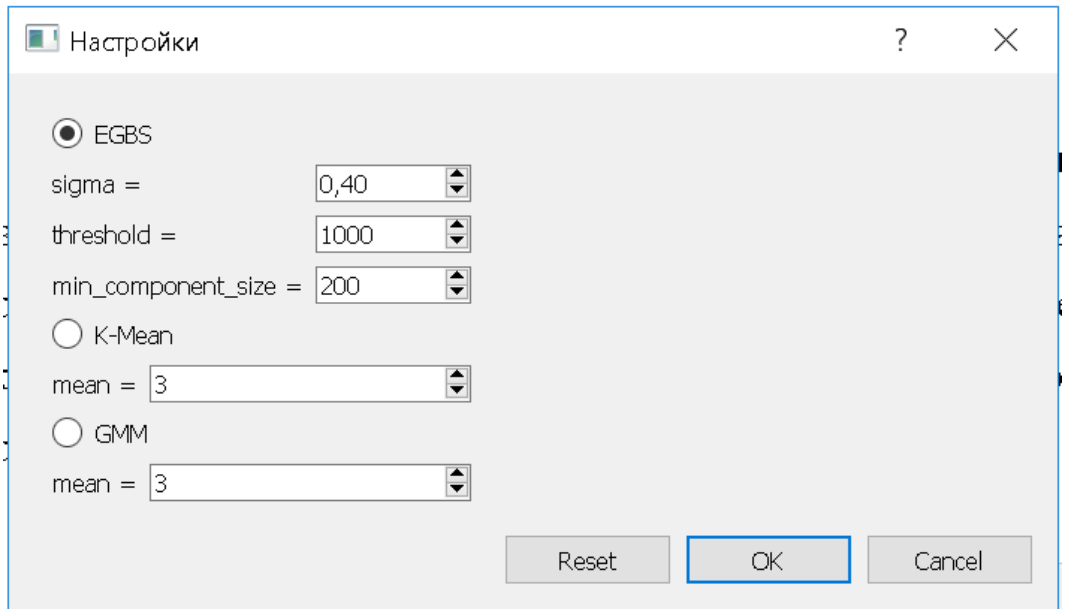

Рисунок 27 – Внешний вид диалогового окна Настройки

Выбрав изображение, алгоритм сегментации и параметры можно запустить сегментацию. По завершению работы алгоритма вместо исходного изображения на главном окно будет показан результат сегментации, на основании которого можно в случае необходимости подкорректировать

параметры и повторить сегментацию, а также сохранить полученное изображение.

Окно отклонение служит для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений, содержит два виджета для загрузки изображений. Одно для загрузки исходного изображения, второе для изображения с выполненной ручной сегментацией. Поддерживается возможность изменения масштаба изображений, выбора алгоритма сегментации и величины выборки. Также можно выбрать метод выделения границ. Структура данного окна изображена на рисунке 28.

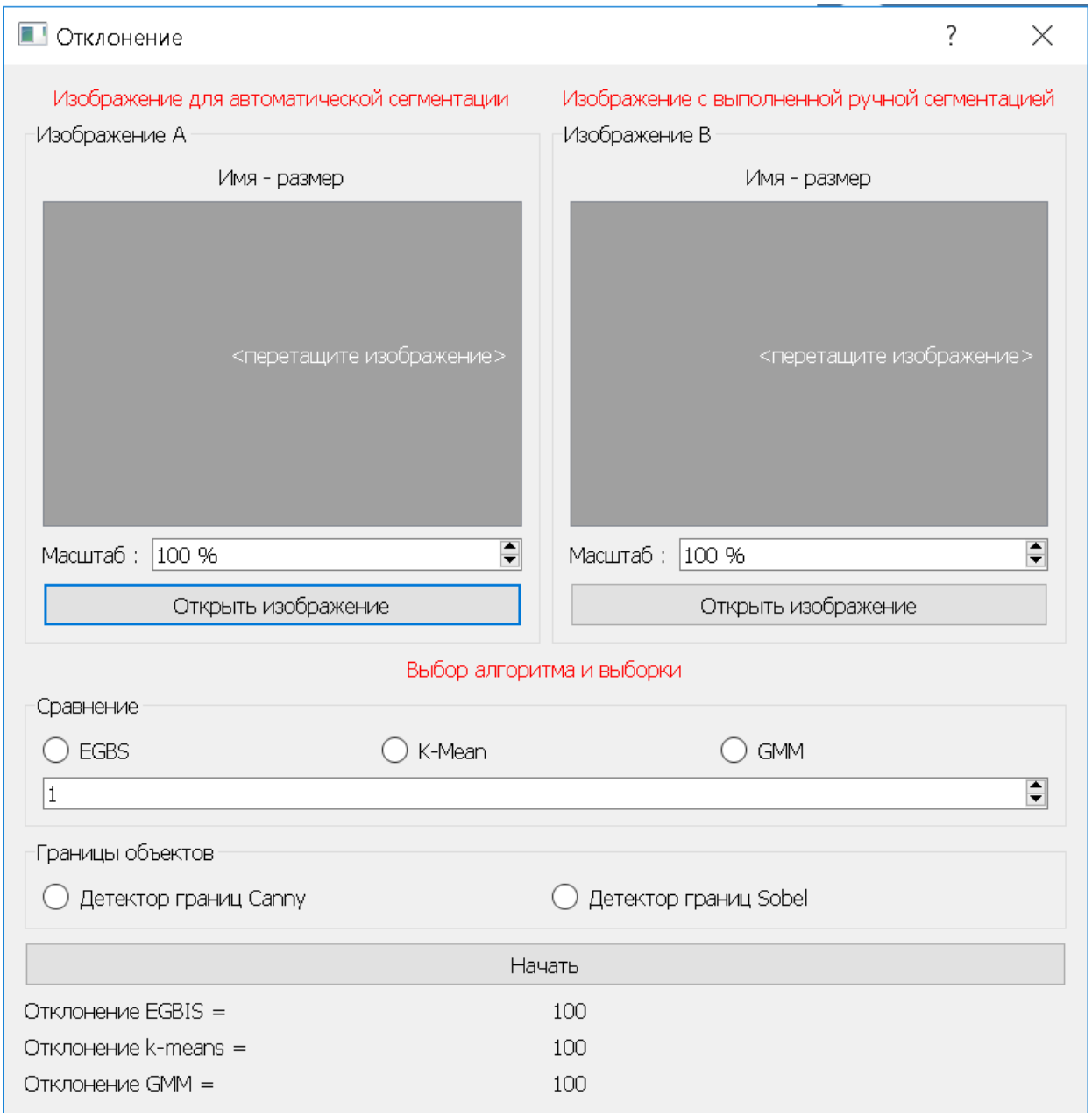

Рисунок 28 – Внешний вид окна Отклонение

5 ОЦЕНКА РАБОТЫ МЕТОДОВ СЕГМЕНТАЦИИ

Для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений необходимо изображение с выполненной ручной сегментацией. Для ручной сегментации использовался графический редактор GIMP. Границы объектов на изображении выделялись линиями со значением RGB = (0, 0, 254), что соответствует красному цвету.

После выполнения выбранного алгоритма сегментации получаем изображение с выделенными разным цветом объектами. Для сравнения необходимо выделить границы этих объектов. Границы всех полученных областей, либо кластеров выделяются с помощью детектора границ Canny. Подготовительными этапами для использования детектора границ являются изменение цветового пространства из RGB в градации серого.

После того как границы объектов определены, границы накладываются на исходное изображение. Значение пикселей являющихся границей объектов на изображении с выполненной автоматической сегментацией в пространстве RGB = (0, 0, 254) Затем происходит анализ границ объектов. Пиксели, представляющие собой границу объектов на изображении с выполненной ручной сегментацией должны так же являться границами объектов на изображении с выполненной автоматической сегментацией. Рассмотрим данный процесс на примере пикселя с координатами (x, y), являющегося границей объекта на изображении с ручной сегментацией. Для оценки отклонения анализируется окно размером 3x3 с центром в точке с координатами (x, y) на изображении с выполненной автоматической сегментацией. Если в окне расположены пиксели границ, то считается, что для данного пикселя отклонение отсутствует. Сравнение границ изображено на рисунке 29.

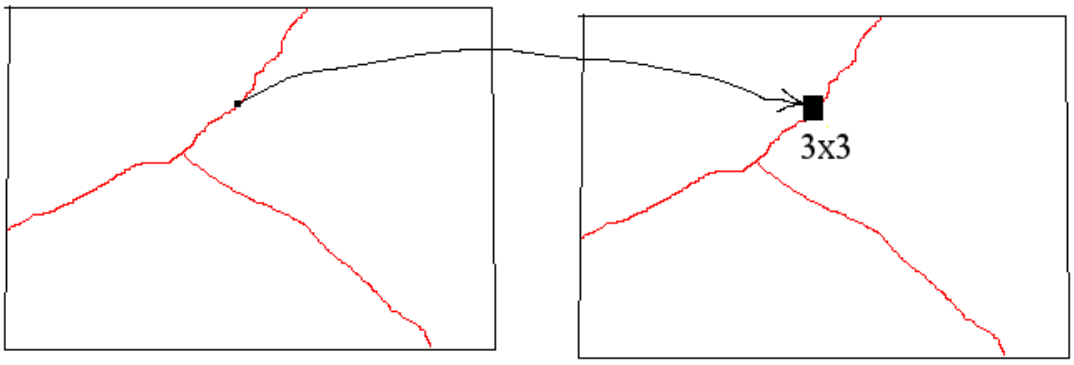

Рисунок 29 – Анализ границ объектов

На основании полученных данных вычисляется отклонение в процентах.

Рассмотрим работу метода на примере изображения:

– исходное изображение (Рисунок 30);

Рисунок 30 – Исходное изображение

– применение алгоритма сегментации (Рисунок 31);

Рисунок 31 – Результат сегментации методом к-средних с количеством кластеров четыре

– выделение границ объектов детектором границ Canny (рисунок 32);

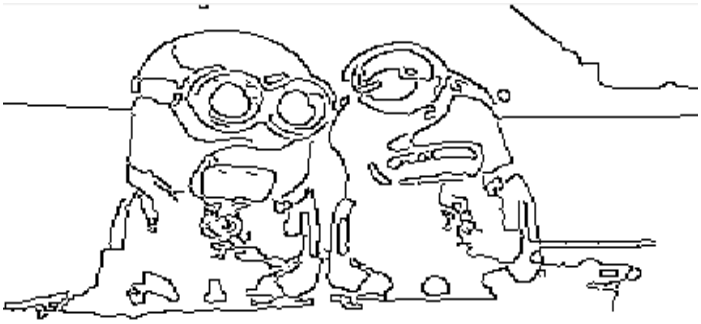

Рисунок 32 – Исходное изображение

– накладываем полученные границы на исходное изображение (Рисунок 33);

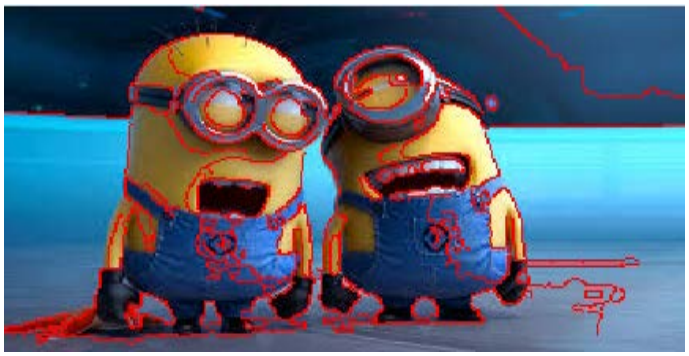

Рисунок 33 – Результат выделения границ

– оценка отклонения результатов ручной и автоматической сегментации изображений (Рисунок 34).

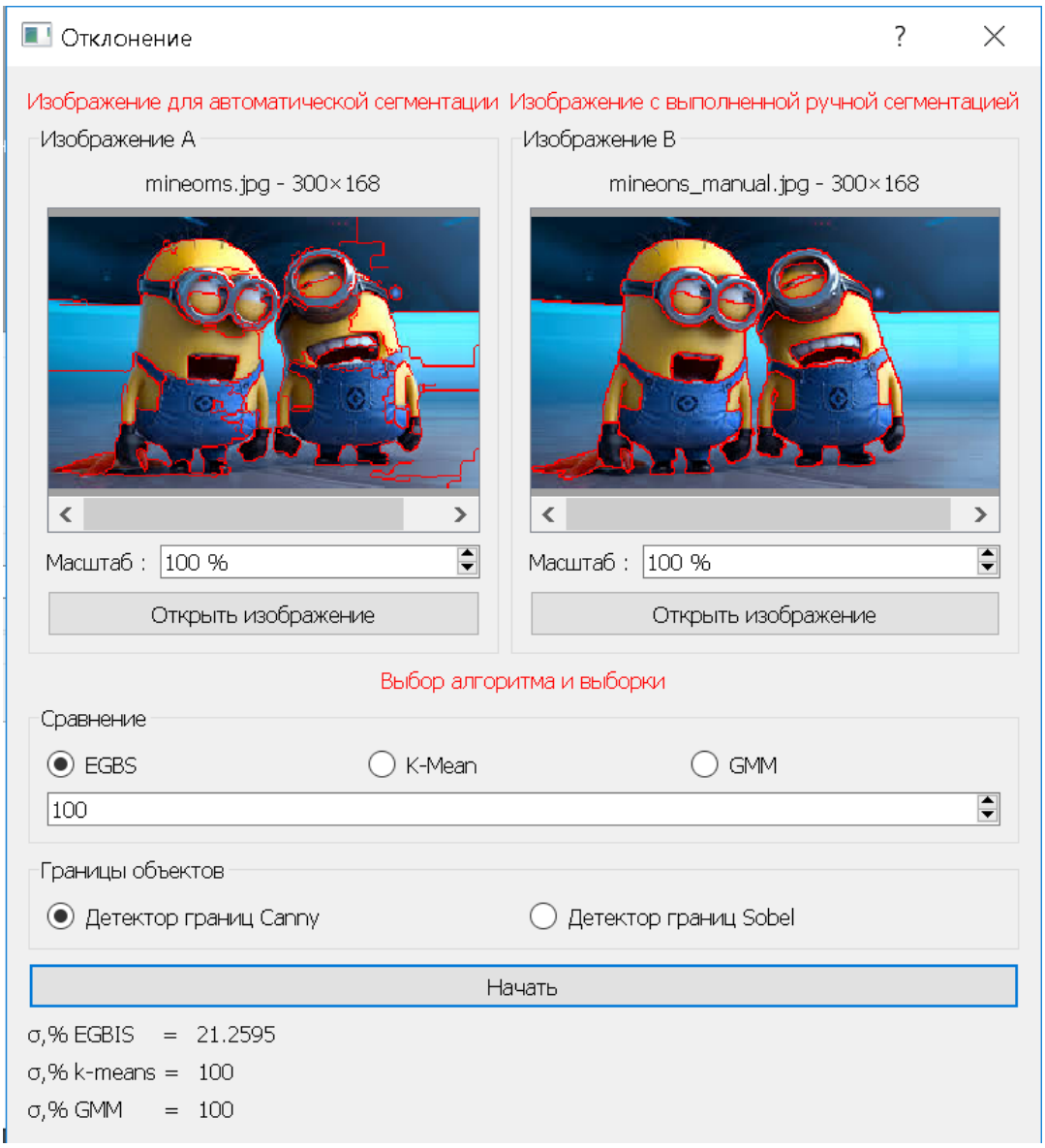

Рисунок 34 – Пример результатов вычисления отклонения для алгоритма эффективной сегментации на графах с выборкой 100

Для получения результатов автоматической сегментации параметры для каждого алгоритма генерируются случайным образом, и оценка с различными параметрами повторяется в зависимости от величины выборки.

В результате оценки на виджете отображается изображение с выделенными границами объектов показавшее наименьшее среднеквадратическое отклонение от границ, выделенных вручную, а также значение среднеквадратического отклонения и величина лишних границ.

Исследование отклонения результатов ручной и автоматической сегментации проводилось на четырех изображениях. Изображения, выбранные для оценки алгоритмов, а также выполненная для них ручная сегментация показаны на рисунке 35, рисунке 36, рисунке 37 и рисунке 38.

Рисунок 35 – Миньоны

Рисунок 36 – Ландшафт

Рисунок 37 – Джордж Мартин

Рисунок 38 – Картина

Результаты исследования приведены в таблице 1.

Таблица 1 – Результаты оценки отклонения

Изображение	Алгоритм	Отклонение, %
Миньоны	Эффективная сегментация изображений на графах	18,7023
	К-средних	26,2366
	Смеси Гаусса	24,1298
Ландшафт	Эффективная сегментация изображений на графах	29,2581
	К-средних	71,6129
	Смеси Гаусса	64,8387
Джордж Мартин	Эффективная сегментация изображений на графах	25,2101

	К-средних	50,8739
	Смеси Гаусса	43,3445
Картина	Эффективная сегментация изображений на графах	47,6744
	К-средних	63,2558
	Смеси Гаусса	45,3488

Наименьшее отклонение показал алгоритм сегментации на графах.

Метод сегментации с помощью смесей Гаусса показал немного большее отклонение чем эффективная сегментация на графах, время необходимое для выполнения сегментации среди всех исследованных алгоритмов самое большое.

Результаты исследования, предоставленные в статье K-Means Cluster Analysis for Image Segmentation показаны в таблице 2.

Таблица 2 – Результаты оценки отклонения

	Цветовое пространство	Правильность,	Точность	Чувствительность	Специфичность
Дерево	L*a*b	0.66	0.54	0.71	0.71
	RGB	0.55	0.54	0.75	0.75
Лошадь	L*a*b	0.71	0.77	0.71	0.71
	RGB	0.53	0.53	0.53	0.53
Голубь	L*a*b	0.57	0.48	0.50	0.77
	RGB	0.56	0.50	0.50	0.76
Мальчик	L*a*b	0.46	0.30	0.33	0.89
	RGB	0.39	0.30	0.30	0.88

Величина отклонение метода к-средних соответствует результатам, полученным в данной работе.

6 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

Задача данного раздела - комплексное описание и анализ экономических аспектов выполненной работы.

Цель данного раздела заключается в том, чтобы оценить полные денежные затраты на разрабатываемый проект, а также дать приближенную экономическую оценку результатов ее внедрения. Это позволяет с помощью традиционных показателей эффективности инвестиций оценить экономическую целесообразность осуществления работы.

6.1 Организация и планирование работ

При организации процесса реализации конкретного проекта необходимо рационально планировать занятость каждого из его участников. А также сроки проведения отдельных работ.

Для наглядности результата планирования работ в данном пункте представлен линейный график реализации проекта. Данный график отображает полный перечень проводимых работ, их исполнители и продолжительность.

Исполнителями являются инженер (И) и научный руководитель (НР). Инженером является исполнитель ВКР.

Перечень работ по написанию ВКР и продолжительность их выполнения представлены в таблице 3.

Таблица 3 – Перечень работ и продолжительность их выполнения

№	Этапы работы	Исполнители	Загрузка исполнителей
1	Постановка целей и задач, получение исходных данных	НР	НР – 100%
2	Составление и утверждение ТЗ	НР, И	НР – 100%, И – 10%

Продолжение таблицы 3

3	Подбор и изучение материалов по тематике	НР, И	НР – 30%, И – 100%
4	Разработка календарного плана	НР, И	НР – 100%, И – 10%
5	Обсуждение литературы	НР, И	НР – 30% И – 100%
6	Разработка алгоритмов приложения	НР, И	НР – 30%, И – 100%
7	Разработка интерфейса приложения	НР, И	НР – 30%, И – 100%
8	Кодирование приложения	НР, И	НР – 10%, И – 100%
9	Отладка и тестирование	НР, И	НР – 10%, И – 100%
10	Оформление пояснительной записки	И	И – 100%
11	Оформление графического материала	И	И – 100%
12	Подведение итогов	НР, И	НР – 60%, И – 100%

6.1.1 Продолжительность этапов работ

Расчет продолжительности этапов работ по написанию данной ВКР проводился при использовании опытно-статистического метода и реализовывался экспертным способом.

Для определения ожидаемых значений продолжительности работ $t_{ож}$ применяется по усмотрению исполнителя одна из двух формул:

$$t_{ож} = \frac{3 \cdot t_{min} + 2 \cdot t_{max}}{5}, \quad (21)$$

$$t_{ож} = \frac{t_{min} + 4 \cdot t_{prob} + t_{max}}{6}, \quad (22)$$

где t_{min} – минимальная продолжительность работы в днях;

t_{max} – максимальная продолжительность работы в днях;

t_{prob} – наиболее вероятная продолжительность работы в днях.

Расчет ожидаемого значения продолжительности работ по каждому этапу проводился с использованием второй формулы, так как она дает более надежные оценки. Исходные данные для расчета $t_{ож}$ по каждому этапу представлены в таблице 4.

Таблица 4 – Исходные данные для расчета $t_{ож}$

№	Этапы работы	t_{min}, дни	t_{max}, дни	t_{prob}, дни	$t_{ож}$
1	Постановка целей и задач, получение исходных данных	1	3	2	2
2	Составление и утверждение ТЗ	1	5	3	3
3	Подбор и изучение материалов по тематике	5	10	7	7,16
4	Разработка календарного плана	2	5	3	3,16
5	Обсуждение литературы	1	3	2	2
6	Разработка алгоритмов приложения	8	15	12	11,83
7	Разработка интерфейса приложения	6	10	8	8
8	Кодирование приложения	8	15	12	11,83
9	Отладка и тестирование	6	10	8	8
10	Оформление пояснительной записки	10	17	12	12,5
11	Оформление графического материала	5	10	8	7,83
12	Подведение итогов	7	12	10	9,83

Для того, чтобы построить линейный график работ, необходимо рассчитать длительность этапов в рабочих днях и перевести полученные значения в календарные дни.

Расчет длительности этапов в рабочих днях производится в соответствии со следующей формулой:

$$\mathrm{Т_{РД}} = \frac{t_{ож}}{\mathrm{К_{ВН}}} \cdot \mathrm{К_{Д}}, \qquad (23)$$

где $t_{ож}$ – продолжительность работы в днях;

$К_{ВН}$ – коэффициент выполнения работ, учитывающих влияние внешних факторов на соблюдение предварительно определенных длительностей, в частности, возможно $К_{ВН} = 1$;

$К_{Д}$ – коэффициент, учитывающий дополнительное время на компенсацию непредвиденных задержек и согласование работ (коэффициент $К_{Д}$ может варьироваться от 1 до 1,2. Конкретное значение коэффициента принимает сам исполнитель).

Исходные данные для расчета длительности этапов в рабочих днях приведены в таблице 5.

Таблица 5 – Исходные данные для расчета длительности этапов в рабочих днях

№	Этапы работы	$t_{ож}$, дни	$К_{ВН}$	$К_{Д}$	$Т_{РД}$
1	Постановка целей и задач, получение исходных данных	2	1	1	2
2	Составление и утверждение ТЗ	3	1	1	3
3	Подбор и изучение материалов по тематике	7,16	1	1	7,17
4	Разработка календарного плана	3,16	1	1	3,16
5	Обсуждение литературы	2	1	1	2
6	Разработка алгоритмов приложения	11,83	1	1,1	13,013
7	Разработка интерфейса приложения	8	1	1,1	8,8
8	Кодирование приложения	11,83	1	1,2	14,196
9	Отладка и тестирование	8	1	1,1	8,8
10	Оформление пояснительной записки	12,5	1	1,1	13,75
11	Оформление графического материала	7,83	1	1	7,83
12	Подведение итогов	9,83	1	1	9,84

Расчет продолжительности этапа в календарных днях осуществляется по следующей формуле:

$$T_{КД} = T_{РД} \cdot T_{К}, \quad (24)$$

где $T_{КД}$ – продолжительность выполнения этапа в календарных днях;

$T_{К}$ – коэффициент календарности, позволяющий перейти от длительности работ в рабочих днях к их аналогам в календарных днях.

Коэффициент календарности $T_{К}$ рассчитывается по следующей формуле:

$$T_{К} = \frac{T_{КАЛ}}{T_{КАЛ} - T_{ВД} - T_{ПД}}, \quad (25)$$

где $T_{КАЛ}$ – календарные дни ($T_{КАЛ}$ = 365);

$T_{ВД}$ – выходные дни ($T_{ВД}$ = 52);

$T_{ПД}$ – праздничные дни ($T_{ПД}$ = 10).

Рассчитаем коэффициент $T_{К}$:

$$T_{К} = \frac{365}{365 - 52 - 10} = 1{,}205.$$

Рассчитаем продолжительность каждого этапа в календарных днях:

$T_{КД1} = 2 \cdot 1{,}205 = 2{,}41;$

$T_{КД2} = 3 \cdot 1{,}205 = 3{,}615;$

$T_{КД3} = 7{,}17 \cdot 1{,}205 = 8{,}64;$

$T_{КД4} = 3{,}16 \cdot 1{,}205 = 3{,}81;$

$T_{КД5} = 2 \cdot 1{,}205 = 2{,}41;$

$T_{КД6} = 13{,}013 \cdot 1{,}205 = 15{,}681;$

$T_{КД7} = 8{,}8 \cdot 1{,}205 = 10{,}604;$

$T_{КД8} = 14{,}196 \cdot 1{,}205 = 17{,}11;$

$T_{КД9} = 8{,}8 \cdot 1{,}205 = 10{,}604;$

$T_{КД10} = 13{,}75 \cdot 1{,}205 = 16{,}569;$

$T_{КД11} = 7{,}83 \cdot 1{,}205 = 9{,}44;$

$T_{КД12} = 9{,}84 \cdot 1{,}205 = 11{,}86;$

Трудозатраты на выполнение проекта для каждого из его участников представлены в таблице 6.

Таблица 6 – Трудозатраты на выполнение проекта

Этап	Ис-пол-ни-тели	Продолжительность работ, дни				Трудоемкость работ по исполнителям чел.-дн.			
						$T_{РД}$		$T_{КД}$	
		t_{min}	t_{max}	t_{prob}	$t_{ож}$	НР	И	НР	И
1	2	3	4	5	6	7	8	9	10
Постановка целей и задач, получение исходных данных	НР	1	3	2	2	2,00	-	2,41	-
Составление и утверждение ТЗ	НР, И	1	5	3	3	3,00	0,3	3,615	0,36
Подбор и изучение материалов по тематике	НР, И	5	10	7	7,16	2,151	7,17	2,592	8,64
Разработка календарного плана	НР, И	2	5	3	3,16	3,16	0,316	3,81	0,381
Обсуждение литературы	НР, И	1	3	2	2	0,6	2	0,723	2,41
Разработка алгоритмов приложения	НР, И	8	15	12	11,83	3,904	13,013	4,7	15,681
Разработка интерфейса приложения	НР, И	6	10	8	8	2,64	8,8	3,181	10,604
Кодирование приложения	НР, И	8	15	12	11,83	1,42	14,196	1,711	17,11
Отладка и тестирование	НР, И	6	10	8	8	0,88	8,8	1,06	10,604
Оформление пояснительной записки	И	10	17	12	12,5	-	13,75	-	16,569
Оформление графического материала	И	5	10	8	7,83	-	7,83	-	9,44
Подведение итогов	НР, И	7	12	10	9,83	5,904	9,84	7,116	11,86
Итого:					87,14	25,66	86,015	30,92	103,659

Исходя из данных, представленных в таблице 5 был составлен линейный график работ (таблица 7).

Таблица 7 – Линейный график работ

Этап	НР	И	Март			Апрель			Май			Июнь	
			10	20	30	40	50	60	70	80	90	100	110
1	2,41	-											
2	3,615	0,36											
3	2,592	8,64											
4	3,81	0,381											
5	0,723	2,41											
6	4,7	15,681											
7	3,181	10,604											
8	1,711	17,11											
9	1,06	10,604											
10	-	16,569											
11	-	9,44											
12	7,116	11,86											

– научный руководитель (НР);

– исполнитель (И).

6.1.2 Расчет накопления готовности проекта

Целью данного пункта является оценка текущих состояний работы над проектом. Величина накопления готовности работы показывает, на сколько процентов по окончании текущего (i-го) этапа выполнен общий объем работ по проекту в целом.

Степень готовности проекта определяется следующей формулой:

$$\text{СГ}_i = \frac{\text{ТР}_i^{\text{Н}}}{\text{ТР}_{\text{общ}}} = \frac{\sum_{k=1}^{i} \text{ТР}_k}{\text{ТР}_{\text{общ}}} = \frac{\sum_{k=1}^{i} \sum_{j=1}^{m} \text{ТР}_{km}}{\sum_{k=1}^{I} \sum_{j=1}^{m} \text{ТР}_{km}}, \qquad (26)$$

где $\text{ТР}_{\text{общ}}$ – общая трудоемкость проекта;

ТР_i (ТР_k) – трудоемкость i-го (k-го) этапа проекта;

$i = \overline{1, I}$ – индекс этапа;

$\text{ТР}_i^{\text{Н}}$ – накопленная трудоемкость i-го этапа проекта по его завершении;

ТР_{ij} (ТР_{ik}) – трудоемкость работ, выполняемых j-м участником на i-м этапе;

$j = \overline{1, m}$ – индекс исполнителя.

Исходя из таблицы 5 величины ТР_{ij} (ТР_{kj}) находятся в девятом и десятом столбцах таблицы. $\text{ТР}_{\text{общ}}$ равна сумме чисел из итоговых ячеек этих столбцов.

Таким образом, степень готовности проекта отражена в таблице 8.

Таблица 8 – Нарастание технической готовности проекта и удельный вес каждого этапа

Этап	ТР_i, %	СГ_i, %
Постановка целей и задач, получение исходных данных	1,79	1,79
Составление и утверждение ТЗ	2,95	4,74
Подбор и изучение материалов по тематике	8,35	13,09
Разработка календарного плана	3,11	16,2
Обсуждение литературы	2,33	18,53
Разработка алгоритмов приложения	15,14	33,67
Разработка интерфейса приложения	10,24	43,91

Кодирование приложения	13,98	57,89
Отладка и тестирование	8,66	66,55
Оформление пояснительной записки	12,31	78,86
Оформление графического материала	7,01	85,87
Подведение итогов	14,1	100

6.1.3 Расчет сметы затрат на выполнение проекта

В состав затрат на создание проекта включается стоимость всех расходов, необходимых для его реализации. Расчет сметной стоимости ее выполнения производится по следующим статьям затрат:

- материалы и покупные изделия;
- заработная плата;
- социальный налог;
- расходы на электроэнергию (без освещения);
- амортизационные отчисления;
- командировочные расходы;
- оплата услуг связи;
- арендная плата за пользование имуществом;
- прочие услуги (сторонних организаций);
- прочие (накладные расходы) расходы.

6.1.4 Расчет затрат на материалы

К данной статье расходов относится стоимость материалов и других материальных ценностей, расходуемых непосредственно в процессе выполнения работ над проектом. Расчет затрат на материалы, использовавшиеся в ходе выполнения проекта, представлены в таблице 9.

Таблица 9 – Расчет затрат на материалы

Наименование материалов	Цена за единицу, руб.	Количество	Сумма, руб.
Qt 5.6.0 коммерческая лицензия	19690,23	1 штука	19690,23
Итого:			19690,23

В нашем случае транспортно-заготовительные расходы (сокращенно ТЗР) отсутствуют. Таким образом, расходы на материалы ($С_{мат}$) составляют 19690,23 рублей.

6.1.5 Расчет заработной платы

Данная статья расходов включает заработную плату научного руководителя и исполнителя проекта, а также премии, входящие в фонд заработной платы.

Среднедневная тарифная заработная плата рассчитывается по следующей формуле:

$$ЗП_{дн-т} = \frac{МО}{24,83}, \quad (27)$$

где МО – величина месячного оклада.

Данная формула учитывает, что в году 298 рабочих дней и, следовательно, в месяце в среднем 24,83 рабочих дня (при шестидневной рабочей неделе).

Для учета в заработной плате премий, дополнительной зарплаты и районной надбавки используется следующий ряд коэффициентов:

$К_{ПР}$ = 1,1;

$К_{доп\ ЗП}$ = 1,188;

$К_{р}$ = 1,3.

Для того, чтобы перейти от тарифной суммы заработка исполнителя проекта к соответствующему полному нужно воспользоваться следующей формулой:

$$ЗП_{полн} = ЗП_{дн-т} \cdot К_{ПР} \cdot К_{доп\ ЗП} \cdot К_{р} = ЗП_{дн-т} \cdot 1{,}699 \qquad (28)$$

Расчет затрат на полную заработную плату представлен в виде таблицы 10. Затраты времени по каждому исполнителю в рабочих днях с округлением до целого взяты из таблицы 6.

Таблица 10 – Расчет затрат на полную заработную плату

Исполнитель	Оклад, руб./мес.	Среднеквартальная ставка, руб./день	Затраты времени, раб. дни	Коэффициент	Фонд заработной платы, руб.
НР	14584,32	587,37	26	1,699	25946,48
И	14874,45	599,05	86	1,699	87529,59
Итого:					113476,07

Таким образом, затраты на заработную плату составили 113476,07 рублей.

6.1.6 Расчет затрат на социальный налог

Затраты на единый социальный налог (сокращенно ЕСН), включающий в себя отчисления в пенсионный фонд, на социальное и медицинское страхование, составляют 30 процентов от полной заработной платы по проекту:

$$С_{соц} = С_{зп} \cdot 0{,}3 \qquad (29)$$

Рассчитаем затраты на ЕСН, используя формулу 1.9:

$$С_{соц} = 113476{,}07 \cdot 0{,}3 = 34042{,}82.$$

Таким образом, затраты на ЕСН составят 34042,82 рублей.

6.1.7 Расчет затрат на электроэнергию

Данный вид расходов включает в себя затраты на электроэнергию, потраченную в ходе выполнения проекта на работу используемого оборудования, рассчитываемые по формуле:

$$С_{эл.об.} = Р_{об} \cdot t_{об} \cdot Ц_э \quad (30)$$

где $Р_{об}$ – мощность, потребляемая оборудованием, кВт;

$Ц_э$ – тариф на 1 кВт·час;

$t_{об}$ – время работы оборудования, час.

Для ТПУ $Ц_Э$ = 5,257 руб./квт·час (с НДС).

Время работы оборудования вычисляется на основе итоговых данных таблицы 6 для инженера ($T_{РД}$) из расчета, что продолжительность рабочего дня равна 8 часов.

$$t_{об} = Т_{РД} \cdot К_t \quad (31)$$

где $К_t \leq 1$– коэффициент использования оборудования по времени, равный отношению времени его работы в процессе выполнения проекта к $T_{РД}$, определяется исполнителем самостоятельно. В рамках выполнения данного проекта коэффициент $К_t$ для ноутбука составил 1.

Рассчитаем время работы ноутбука: $t_{об} = 86{,}015 \cdot 1 \cdot 8 = 688$.

Мощность, потребляемая оборудованием, рассчитывается по формуле 20:

$$P_{ОБ} = P_{ном} \cdot К_С \quad (33)$$

где $Р_{ном}$ – номинальная мощность оборудования в киловаттах;

$К_С$ – коэффициент загрузки, зависящий от средней степени использования номинальной мощности. Для технологического оборудования малой мощности $К_С$ равен единице.

Номинальная мощность оборудования, используемого в ходе реализации проекта для персонального компьютера составляет – 0,085 киловатт.

Расчет затрат на электроэнергию для технологических целей представлен в таблице 11.

Таблица 11 – Затраты на электроэнергию технологическую

Наименование оборудования	Время работы оборудования $t_{об}$, час	Потребляемая мощность оборудования $P_{ОБ}$, кВт	Затраты, руб.
Ноутбук	688	0,085	307,43
Итого:			307,43

Таким образом, затраты на электроэнергию в процессе выполнения проекта составили 307,43 рублей.

6.1.8 Расчет амортизационных расходов

В статье «Амортизационные отчисления» рассчитывается амортизация используемого оборудования за время выполнения проекта.

Расчет амортизационных расходов осуществляется по формуле 34.

$$\mathrm{С_{АМ}} = \frac{\mathrm{Н_А} \cdot \mathrm{Ц_{ОБ}} \cdot t_{\mathrm{рф}} \cdot n}{F_{\mathrm{Д}}}, \qquad (34)$$

где $\mathrm{Н_А}$ – годовая норма амортизации единицы оборудования;

$\mathrm{Ц_{ОБ}}$ – балансовая стоимость единицы оборудования с учетом ТЗР;

$\mathrm{F_Д}$ – действительный годовой фонд времени работы соответствующего оборудования;

$t_{\mathrm{рф}}$ – фактическое время работы оборудования в ходе выполнения проекта;

n – число задействованных однотипных единиц оборудования.

Годовая норма амортизации рассчитывается по следующей формуле:

$$\mathrm{Н_А} = \frac{1}{\mathrm{СА}}, \qquad (35)$$

где СА – срок амортизации оборудования.

Из постановления правительства «О классификации основных средств, включенных в амортизационные группы» срок амортизации для электронно-вычислительной техники, составляет свыше двух лет до трех лет включительно. Зададим значение срока амортизации, равное 2,5 года.

Рассчитаем годовую норму амортизации:

$$Н_А = \frac{1}{2,5} = 0,4.$$

Таким образом, годовая норма амортизации для ноутбука составляет 0,4.

Действительный годовой фонд времени работы оборудования можно принять как произведение количества рабочих дней в году на продолжительность использования оборудования в течение одного рабочего дня (в часах).

Если учесть, что в году 298 рабочих дней и восьмичасовой рабочий день, то действительный годовой фонд времени работы ноутбука составляет:

$F_Д = 298 \cdot 8 = 2384$ часа.

Расчет амортизационных расходов представлен в таблице 12.

Таблица 12 – Амортизационные расходы на оборудование

Оборудование	Годовая норма амортизации $Н_А$	Балансовая стоимость $Ц_{ОБ}$, руб.	Фактическое время работы $t_{рф}$, часы	Действительный годовой фонд времени работы, часы	Амортизационные расходы $С_{АМ}$, руб.
Ноутбук	0,4	47900,00	688	2384	5529,4
Итого:					5529,4

Таким образом амортизационные расходы на оборудование, используемое в ходе реализации проекта, составили 5529,4 рубля.

6.1.9 Расчет прочих расходов

В статье «Прочие расходы» отражены расходы на выполнение проекта, которые не учтены в предыдущих статьях. Данный тип расходов рассчитывается по следующей формуле:

$$С_{проч} = (С_{мат} + С_{ЗП} + С_{соц} + С_{эл.об.} + С_{АМ}) \cdot 0,1, \qquad (36)$$

Рассчитаем затраты на прочие расходы при выполнении проекта:

$С_{проч}$ = (19690,23 + 113476,07 + 34042,82 + 307,42 + 5529,4) · 0,1;

$С_{проч}$ = 17304,59.

Таким образом, затраты на прочие расходы при выполнении дипломного проекта составили 17304,59 рубля.

6.1.10 Расчет общей себестоимости разработки

Проведя расчет по всем статьям сметы затрат на разработку, можно определить общую себестоимость проекта «Приложение для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений» (таблица 13).

Таблица 13 – Смета затрат на разработку проекта

Статья затрат	Условное обозначение	Сумма, руб.
Материалы и покупные изделия	$С_{мат}$	19690,23
Основная заработная плата	$С_{зп}$	113476,07
Отчисления в социальные фонды	$С_{соц}$	34042,82
Расходы на электроэнергию	$С_{эл.об.}$	307,42
Амортизационные отчисления	$С_{АМ}$	5529,4
Прочие расходы	$С_{проч}$	17304,59
Итого:		190350,53

Таким образом, затраты на разработку составили 190350,53 рублей.

6.1.11 Расчет прибыли

Так как недостаточно данных для применения более точных методов расчета прибыли от реализации проекта, примем прибыль как двадцать процентов от полной себестоимости проекта.

Рассчитаем объем прибыли от реализации проекта:

С пр = 190350,53 · 0,2 = 38070,106.

Исходя из полученных расчетов, прибыль от реализации проекта составляет 38070,106 рубля.

6.1.12Расчет НДС

НДС составляет восемнадцать процентов от суммы затрат на разработку и прибыли.

Рассчитаем НДС:

$С_{НДС}$ = (190350,53 + 38070,106) · 0,18 = 41115,72.

Таким образом, НДС составляет 41115,72 рублей.

6.1.13 Цена разработки НИР

Цена равна сумме полной себестоимости, прибыли и НДС, в нашем случае:

$Ц_{НИР}$ = 190350,53 + 38070,106 + 41115,72 = 269536,35.

6.2 Оценка экономической эффективности

В связи с тем, что алгоритмы, реализованные в приложении, имеют множество областей применения, а также в связи с недостатком информации для полноценного анализа, полная оценка экономической эффективности проекта невозможна.

Разработанное приложение выполняет сегментацию цифровых изображений различными методами, а также оценивает качество сегментации по отношению к результатам ручной сегментации.

Один из возможных вариантов использования приложения – увеличение контрастности и удалением шумов. Рассчитаем экономическую эффективность

данного сценария, связанную с увеличением количества принимаемых пациентов врачом рентгенологом на примере ОГАУЗ «Томская районная больница».

В зависимости от объекта интереса стоимость рентгенографии варьируется от 150 до 1200 рублей. Возьмем стоимость в 275 рублей. В среднем в день рентгенолог принимает 10 пациентов. При таких параметрах выручка за год составит 819500 рублей без НДС и 967010 рублей с НДС.

Средняя заработная плата рентгенолога составляет 20000 рублей. За год больница начисляет врачам сумму в размере:

$С_{ЗП} = 20000 \cdot 12 \cdot 2 = 480000.$

Затраты на социальный налог:

$С_{соц} = 480000 \cdot 0,3 = 144000$

Затраты на содержание помещения невозможно посчитать из-за недостаточного количества данных.

Таким образом, за один год больница тратит на оплату труда рентгенологов 624000 рублей. И прибыль составит 195500 рублей.

Применив алгоритм смеси гауссовых распределений для уменьшения шумов и увеличения контрастности, помимо более точных диагнозов сократит время, которое тратит врач на постановку диагноза и позволит принять большее количество пациентов. Скорость работы алгоритма по приблизительным оценкам сократит на 20 % время которое врач рентгенолог тратит на одного пациента. Таким образом ежедневно количество принятых пациентов увеличится на 2 и составит 12 человек. Выручка за год после внедрения приложения составит 983400 рублей без НДС и 1160412 рублей с НДС. Прибыть составит 359400 рублей.

Рассчитаем на сколько увеличится прибыль:

359400 – 195500 = 163900.

Таким образом ОГАУЗ «Томская районная больница» используя разработанное приложение увеличит прибыль на 163900рублей в год.

Срок окупаемости проекта составляет 1,6.

6.3 Оценка научно-технического уровня НИР

Научно-технический уровень характеризует влияние проекта на уровень и динамику обеспечения научно-технического прогресса в данной области. Для оценки научной ценности, технической значимости и эффективности использовался метод балльных оценок.

Сущность метода заключается в том, что на основе оценок признаков работы определяется интегральный показатель ее научно-технического уровня, рассчитываемый по следующей формуле:

$$I_{\text{НТУ}} = \sum_{i=1}^{3} R_i \cdot n_i \qquad (37)$$

где $I_{\text{НТУ}}$ – интегральный индекс научно-технического уровня;

R_i – весовой коэффициент i-го признака научно-технического эффекта;

n_i – количественная оценка i-го признака научно-технического эффекта в баллах.

Оценки научно-технического уровня данной работы представлены в виде таблицы 14.

Таблица 14 – Оценка научно-технического уровня НИР

Значимость	Фактор НТУ	Уровень фактора	Выбранный балл	Обоснование выбранного балла
0,4	Уровень новизны	Относи-тельно новая	3	Облегчает выбор алгоритма сегментации
0,1	Теоретический уровень	Разработка программы	7	Разработка программы со множеством алгоритмов для различных задач
0,5	Возможность реализации	В течение первых лет	10	Приложение сразу готово к использованию

Используя данные таблицы 14 рассчитаем показатель научно-технического уровня для данного проекта:

$I_{НТУ} = 0,4 \cdot 3 + 0,1 \cdot 7 + 0,5 \cdot 10 = 6,9.$

В таблице 15 указано соответствие качественных уровней НИР значениям показателя, рассчитываемого по формуле 24.

Таблица 15 – Соответствие качественных уровней НИР значению интегрального индекса научно-технического уровня

Уровень НТЭ	Показатель НТЭ
Низкий	1-4
Средний	4-7
Высокий	8-10

Исходя из данных, указанных в таблице 15, данный проект имеет средний уровень научно-технического эффекта.

7 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Данный раздел представляет собой анализ деятельности и последствий использования разработанного приложения. А также содержит необходимые рекомендации по работе и описание действий для снижения негативных последствий.

Согласно ГОСТ Р ИСО 26000-2012 [20] социальная ответственность – ответственность организации за воздействие её решений и деятельности на общество и окружающую среду через прозрачное и этичное поведение, которое содействует устойчивому развитию, включая здоровье и благосостояние общества; учитывает ожидания заинтересованных сторон; соответствует применяемому законодательству и согласуется с международными нормами поведения; интегрировано в деятельность всей организации и применяется в её взаимоотношениях.

Разработанное приложение выполняет сегментацию цифровых изображений 3 методами:

- эффективная сегментация на графах;
- к-средние;
- смеси гауссовых распределений.

А также вычисляет отклонение результатов сегментации этих алгоритмов от результатов ручной сегментации, которую мы принимает за эталон.

7.1 Анализ возможных сбоев и их последствий

Сегментация изображений применяется во многих областях:

- выделение объектов на спутниковых снимках;
- машинное зрение;
- идентификация лиц;
- идентификация отпечатков пальцев;
- управление дорожным движениям;
- медицинские изображения.

Сегментация с использованием алгоритмов реализованных в приложении, требует ввода дополнительных параметров, что автоматически подразумевает возможность ошибки, особенно если у пользователя приложения недостаточно опыта или знаний в данной области. Ошибки могут привести к ужасным происшествиям.

В первую очередь ошибок при сегментации стоит избегать при работе с медицинскими изображениями. Некоторые варианты использования в медицине:

- обнаружение опухолей и патологий
- определение объёмов тканей
- хирургия с использованием компьютера
- диагностика
- изучение анатомии

Ошибки при обнаружении опухолей и патологий могут привести к летальному исходу пациента, а также материальным расходам больницы из-за халатности специалиста. Такие же последствия следуют при не правильном планировании лечения, диагностировании и определении объемов тканей.

Далее, ошибки при выделении объектов на спутниковых снимках при использовании в военных областях могут привести к жертвам среди мирного населения.

Варианты ошибок при использовании сегментации в системах управления дорожным движениям могут иметь различный характер. Например, контролирование скорости автотранспорта. В данном случае возможен вариант, когда по ошибке штраф за превышение скорости был отправлен не тому человеку, или же факт нарушения не был замечен вовсе.

Рассмотрим пример последствия от ошибки, когда произошел сбой при распознавании отпечатков пальцев. Такая ситуация вероятнее всего может произойти с разного рода правоохранительными органами преимущественно в ходе расследования происшествия, либо осуществлении проверки.

Последствиями могут быть как нанесение психологической травмы человеку, так и вред здоровью, а также преступник может остаться безнаказанным.

7.2 Безопасное конструирование программного обеспечения

Разработка приложения – это сложный процесс, который включает в себя множество компонентов.

Согласно Макконелу [21] в общем виде конструирование состоит из следующих этапов:

- определение проблемы;
- выработка требований;
- создание плана конструирования;
- разработка архитектуры;
- детальное проектирование;
- кодирование и отладка;
- блочное тестирование;
- интеграционное тестирование;
- интеграция;
- тестирование системы;
- корректирующее сопровождение.

Конструирование имеет такое большое значение, так как именно на этом этапе определяются требования к приложению и его архитектура. Правильное конструирование гарантирует эффективность приложения.

В данном проекте использовался итеративный подход к разработке. То есть все работы выполнялись с анализом результата и корректированием предыдущих этапов. Данный подход был выбран в связи с тем, что при нем затраты на исправление дефектов обычно ниже, а также различные возможные ошибки выявляются раньше.

7.3 Безопасность пользовательского интерфейса

Интерфейс - система средств и правил, регламентирующая и обеспечивающая взаимодействие нескольких процессов или объектов.

Пользовательский интерфейс приложения представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с ним.

Для корректной работы приложения интерфейс пользователя должен интуитивно понятным, а также обеспечивать защиту от различных ошибок, которые может совершить пользователь, например, от ошибок ввода данных.

Возможной ошибкой пользователя при работе может быть попытка загрузить в приложение файл неверного формата или же файл, не являющийся изображением.

Реакция на подобное действие изображена на рисунке 39.

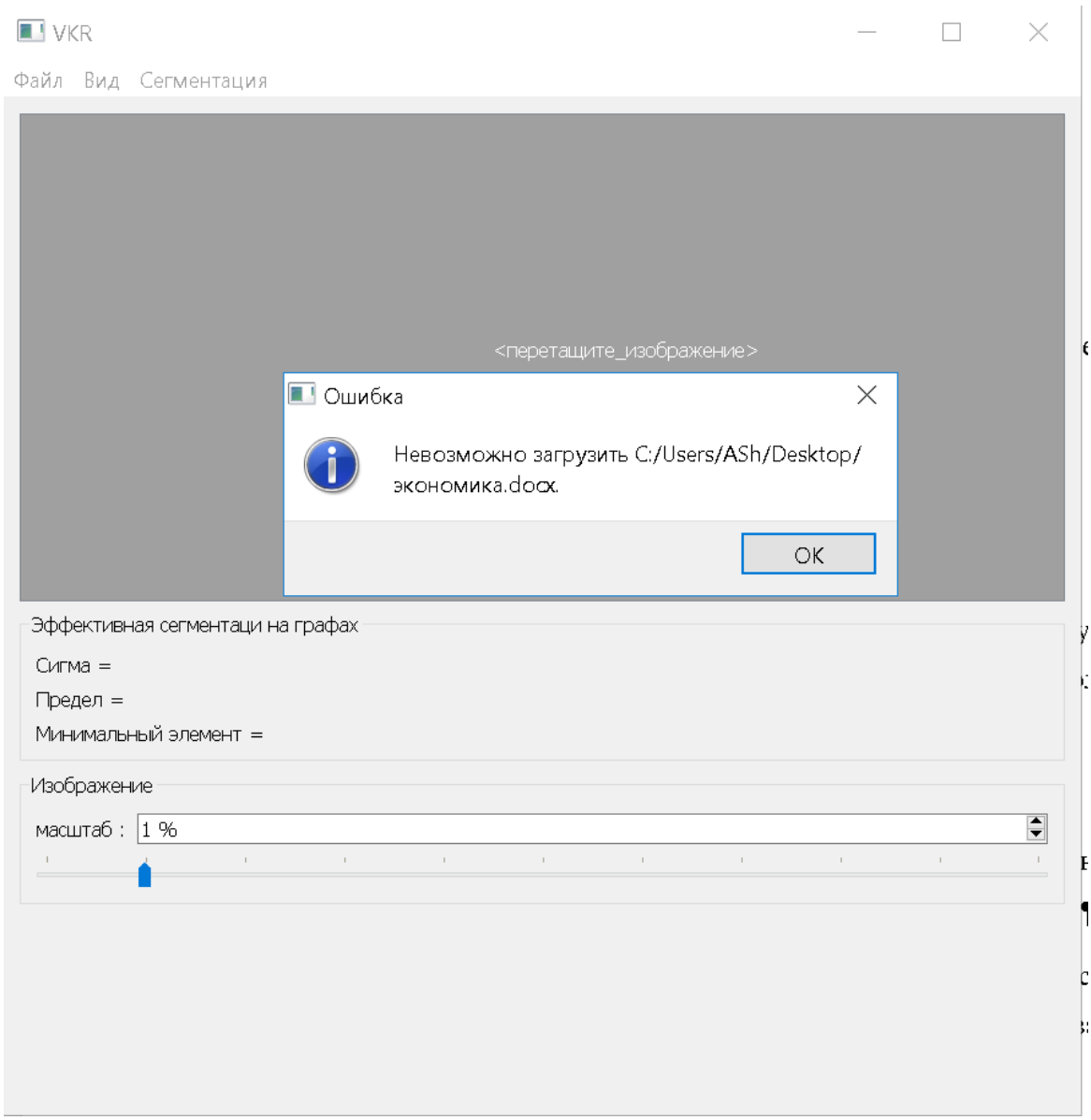

Рисунок 39 – Ошибка файла

Как видно из рисунка пользователь по ошибке выбрал текстовый файл, реакцией приложения является сообщение пользователю о том, что выбран файл неверного формата, работа приложения продолжается в обычном режиме и после нажатия на кнопку ОК можно выбрать другой файл.

Также обеспечена валидация ввода данных для алгоритмов. Все поля ввода параметров имеют строгие ограничения и не позволяют ввести недопустимые значения, отрицательные значения или же символы.

Однако данный метод защиты предотвращает ввод только очевидно неподходящих данных и не обеспечивает полной защиты от ввода ошибочной информации. Так, например, используя метод к-средних или алгоритм смесей

гауссовых распределений пользователь может выбрать количество сегментов, не подходящее для данного изображения.

Кнопки выполнения сегментации, а также изменения масштаба изображения становятся активными только после того как пользователь загрузит в приложение интересующее его изображение.

7.4 Проблемы смесей гауссовых распределений

В связи с наличием особенностей сегментации изображений используя алгоритм смесей гауссовых распределений рассмотрим ошибку, связанную с максимальными рамками правдоподобия.

Для простоты рассмотрим гауссовы смеси, кластеры которого имеют ковариационные матрицы:

$$\sum_{к} = I \cdot \sigma^2, \qquad (38)$$

где I – единичная матрица, хотя выводы справедливы и для других типов матриц ковариаций.

Матрицы ковариаций делятся на:

– сферическая;

– диагональная;

– полная.

Предположим, что один из элементов модели, например jый компонент, имеет центр кластера μ_j, который точно соответствует некоторой точке из набора данных.

Функция правдоподобия для этой точки примет вид:

$$N\left(x_n \middle| x_n, \sigma_j^2 I\right) = \frac{1}{(2\pi)^{1/2}} \frac{1}{\sigma_j} \qquad (39)$$

Если мы рассмотрим предел $\sigma_j \to 0$, то функция правдоподобия стремится к бесконечности. Таким образом, максимизация функции логарифмического правдоподобия не является хорошо проработанной задачей, потому что такие особенности всегда будут присутствовать и будут происходить

всякий раз, когда один из кластеров разрушается в специфической точке. Эта проблема не возникает в случае одного гауссова распределения. Однако, как только у нас есть как минимум два кластера в смесях, один из кластеров может иметь конечную дисперсию и присвоить конечное значение вероятности для всех точек, в то время как другой может сократится в специфической точке и тем самым продолжить увеличивать значение логарифмического правдоподобия. Это изображено на рисунке 40.

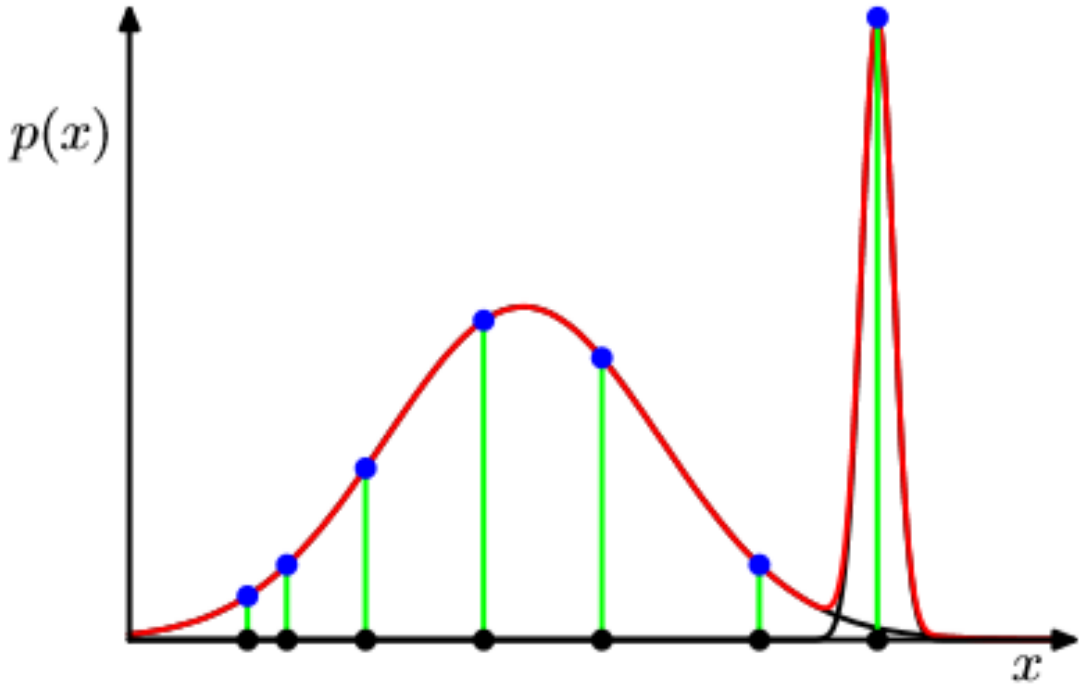

Рисунок 40 – Особенности функции правдоподобия

При применении данного алгоритма были приняты действия по устранению таких паталогических ситуаций. Чтобы избежать подобных особенностей алгоритма было принято решение отслеживать, когда кластер разрушается и сбрасывать его значение до случайной величины в пределах от 0 до 255. Так же обновляется матрица ковариаций данного кластера и затем продолжается оптимизация.

7.5 Безопасность оборудования

Поскольку разрабатываемый проект представляет собой приложение, то логично, что воздействие факторов, влияющих на физическую платформу, на

которой установлено приложение, невозможно предотвратить средствами самого приложения. Для защиты от таких факторов необходимо обеспечить платформу физическими средствами, которые позволят игнорировать данные факторы.

Для кратковременного питания компьютера и его устройств при неполадках в сети применяется источник бесперебойного питания (ИБП).

Согласно ГОСТ 13109-97 [22] нормы в электропитающей сети:

- напряжение 220 В ± 5 % ;
- частота 50 Гц ± 0,2 Гц;
- коэффициент нелинейных искажений формы напряжения менее 8 % (длительно) и менее 12 % (кратковременно).

ИБП так же используется для корректировки параметров электрического тока при отклонении его параметров от нормы.

Не все устройства нуждаются в ИБП. Так ноутбук и другие устройства, имеющие встроенную аккумуляторную батарею при неполадках в сети, могут работать еще некоторое время.

Характеристики ИБП:

- выходная мощность;
- выходное напряжение;
- время переключения;
- время автономной работы;
- ширина диапазона сетевого напряжения;
- срок службы аккумуляторных батарей.

Внешний вид ИБП изображен на рисунке 41.

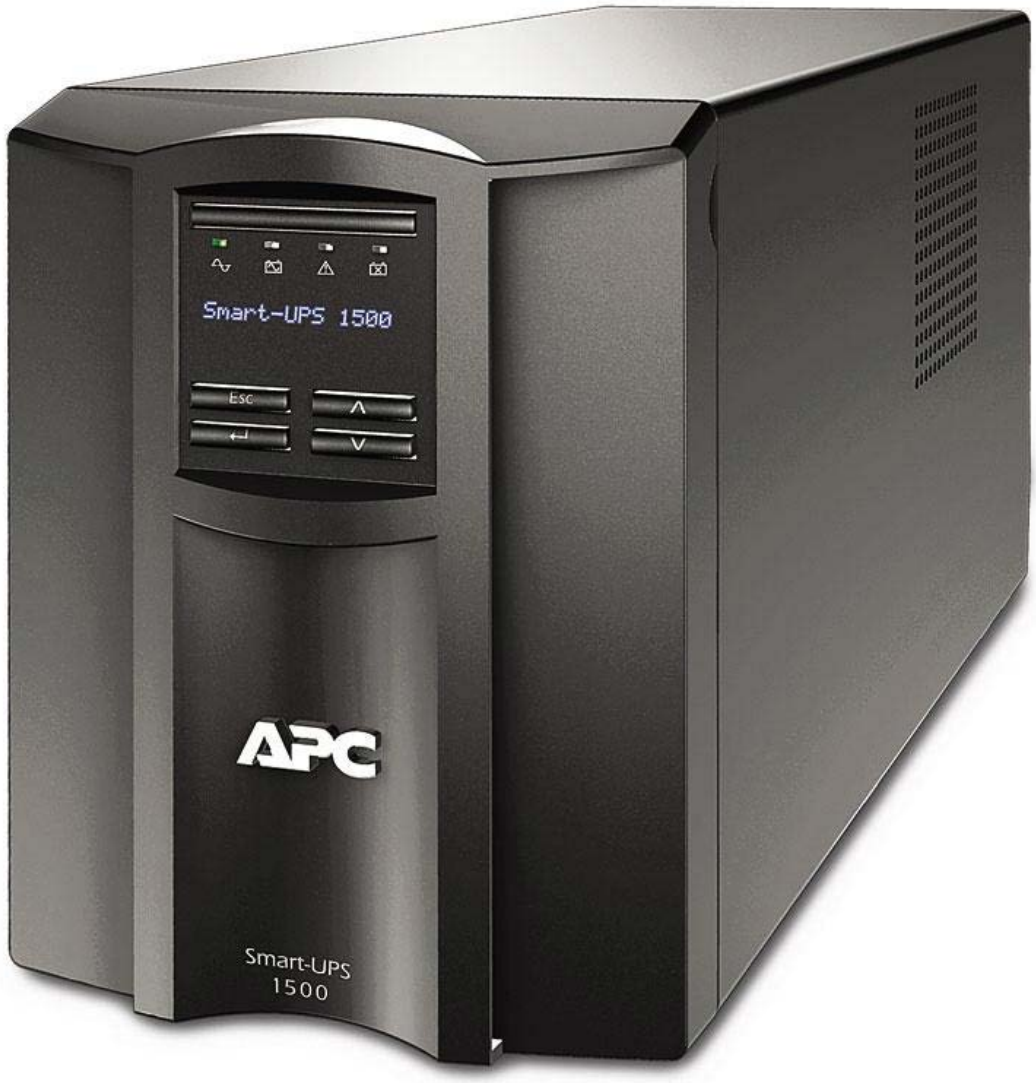

Рисунок 41 – Внешний вид ИБП

Но источник бесперебойного питания подходит только для предотвращения кратковременных неполадок в сети. Для долговременного питания компьютера лучше всего подходит источник резервного питания (ИРП).

7.6 Технические характеристики

Технические характеристики оборудования, на котором протестирована работоспособность приложения:

- процессор Intel Core i7-36015QM 2,30 ГГц;
- ОЗУ 8 ГБ;
- 64 разрядная система;
- операционная система Windows 10;

– видеокарта NVIDIA GeForce GTX 850M.

Для достоверности приложение было протестировано еще на двух персональных компьютерах.

Минимальные характеристики на которых была проверена работоспособность:

– процессор Intel Core i3-2310M 2,10 ГГц;

– ОЗУ 4 ГБ;

– видеокарта AMD Radeon HD 6630M

Но работа приложения возможно и с более низкими характеристиками.

Также была протестирована работа приложения с другими операционными системами. Результаты тестирования показали полную совместимость с операционными системами Windows 7 и Windows 8.

7.7 Требования к персоналу

В связи с опасными последствиями ошибок персонала использующего приложение требуется владеть не только достаточными знаниями в своей области, но и иметь знания в области сегментации цифровых изображений, алгоритмов, а также иметь опыт для выбора подходящих параметров.

Так, например, необходимым требованием для использования приложения при сегментации медицинских изображений является наличие у персонала высшего медицинского образования.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы было проведено исследование алгоритмов сегментации и реализовано приложение для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений.

Были исследованы отклонения результатов сегментации алгоритма эффективной сегментации изображений основанный на графах, алгоритма к-средних и смесей Гаусса.

Быстрее всего выполнял сегментацию алгоритм эффективной сегментации на графах, а также показал наименьшее отклонение.

Небольшое отклонение также показал алгоритм сегментации смесями Гаусса. Но у данного алгоритма есть узкое место – это скорость выполнения. Из-за того, что для инициализации используется алгоритм к-средних и трехмерное пространства в функциях данный алгоритм выполнял сегментацию дольше всего.

В ходе написания дипломного проекта в разделе финансового менеджмента, ресурсоэффективности и ресурсосбережения была проанализирована экономическая эффективность разработанного приложения и стоимость разработки. В разделе социальная ответственность были описаны возможные сбои и ошибки, а также принятые меры и рекомендации по предотвращению подобных ситуаций.

Разработанное приложение позволяет оценить отклонение результатов сегментации и от специфики изображений, области применения того или иного алгоритма сегментации сделать выводы о приемлемости его использования.

CONCLUSION

During execution of final qualifying work was a study of segmentation algorithms and implemented an application for assessment of deviations of the results of manual and automatic segmentation of digital images.

Deviation of the results of segmentation algorithm efficient graph-based image segmentation, algorithm k-means and Gaussian mixtures were researched.

The fastest way to perform segmentation is efficient graph-based image segmentation, and also showed the greatest deviation for most images.

A small deviation showed segmentation algorithm Gaussian mixture model. But this algorithm bottleneck is the speed of execution. Due to the fact that the algorithm used to initialization the k-means and 3-dimensional space to perform functions of the segmentation algorithm is the longest.

During the writing of the graduation project under the financial management, resource efficiency and resource analyzed the economic efficiency of the developed application and development cost. In the social responsibility were described possible failures and errors, as well as the measures and recommendations to prevent similar situations.

The developed application allows to evaluate the deviation of the segmentation results and images from the specifics of the application of a segmentation algorithm to draw conclusions about the acceptability of its use.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Felzenszwalb Pedro F. Efficient Graph-Based Image Segmentation [Электронный ресурс] // Pedro F. Felzenszwalb, Daniel P. Huttenlocher. URL: https://www.cs.cornell.edu/~dph/papers/seg-ijcv.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 17.03.2016 г.

2) Szeliski R. Computer Vision: Algorithms and Applications [Электронный ресурс] URL: http://szeliski.org/Book, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 20.04.2016 г.

3) Christopher M. Bishop Pattern Recognition and Machine Learning Springer 2007. – 739 p.

4) Rother C. GrabCut -Interactive Foreground Extraction using Iterated Graph Cuts [Электронный ресурс] / C. Rother, V. Kolmogorov, A. Blake URL: http://research.microsoft.com/apps/pubs/default.aspx?id=67890, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 11.05.2016 г.

5) Mikulka J. Evaluation of Errors in Manual Image Processing [Электронный ресурс] / J. Mikulka, E. Gescheidtov´a, and K. Bartu˘sek. // PIERS Proceedings. 202–204, Xian, China, March 22–26, 2010. URL: http://www.utee.feec.vutbr.cz/files/mikulka/publikace/2011_Evaluation_errors_manual.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 15.04.2016 г.

6) Bazille A. Impact of semiautomated versus manual image segmentation errors on myocardial strain calculation by magnetic resonance tagging [Электронный ресурс] URL: http://www.ncbi.nlm.nih.gov/pubmed/8034448, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 20.04.2016 г.

7) Tatiraju S. Image Segmentation using k-means clustering, EM and Normalized Cuts [Электронный ресурс] / Suman Tatiraju, Avi Mehta URL: http://www.ics.uci.edu/~dramanan/teaching/ics273a_winter08/projects/avim_report.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 15.04.2016 г.

8) Aqil B. K-Means Cluster Analysis for Image Segmentation[Электронный ресурс] / S. M. Aqil Burney, Humera Tariq URL:

http://research.ijcaonline.org/volume96/number4/pxc3896360.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 17.05.2016 г.

9) Pantofaru C. A. Comparison of Image Segmentation Algorithms [Электронный ресурс] / C. Pantofaru, M. Hebert // The Robotics Institute Carnegie Mellon University Pittsburgh. September 1, 2005. URL: https://www.ri.cmu.edu/pub_files/pub4/pantofaru_caroline_2005_1/pantofaru_caroline_2005_1.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 10.03.2016 г.

10) Fukunaga K. The estimation of the gradient of a density function, with applications in pattern recognition [Электронный ресурс] // K. Fukunaga ; L. Hostetler. URL: http://ieeexplore.ieee.org, ограниченный . – Загл. с экрана. – Яз. англ. Дата обращения: 5.05.2016 г.

11) Unnikrishman R. A measure for Objective Evaluation of Image Segmentation Algorithms [Электронный ресурс] / R. Unnikrishman, C. Pantofaru. M. Hebert // CVPR workshop on Emperical Evaluation Methods in Computer Vision., 2005. URL: https://www.willowgarage.com/sites/default/files/CVPR%202005%20-%20A%20Measure%20for%20Objective%20Evaluation.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 21.03.2016 г.

12) Unnikrishman R. Measures of Similarity [Электронный ресурс] / R. Unnikrishman, M. Hebert. URL: https://www-preview.ri.cmu.edu/pub_files/pub4/unnikrishnan_ranjith_2005_1/unnikrishnan_ranjith_2005_1.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 26.03.2016 г.

13) Martin D. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics [Электронный ресурс]/ D. Martin, C. Fowlkes, D. Tal, J. Malik, // ICCV, July 2001. URL: http://www.ics.uci.edu/~fowlkes/papers/mftm-iccv01.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 02.05.2016 г.

14) Yager R. R. Approximate Clustering Via the Mountain Method[Электронный ресурс]/ Ronald R. Yager, Dimitar P. Filev. URL: http://www.polytech.univ-savoie.fr/fileadmin/polytech_autres_sites/sites/listic/busefal/Papers/58.zip/58_01.pdf. – Загл. с экрана. – Яз. англ. Дата обращения: 01.02.2016 г.

15) Canny J. A Computational Approach To Edge Detection [Электронный ресурс] URL: http://cmp.felk.cvut.cz/~cernyad2/TextCaptchaPdf/A%20Computational%20Approach%20to%20Edge%20Detection.pdf, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 17.05.2016 г.

16) Руководство Microsoft по проектированию архитектуры приложений, 2-е издание [Электронный ресурс] URL: https://blogs.msdn.microsoft.com/rusaraford/2010/02/16/microsoft-1082, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 09.05.2016 г.

17) Шлее М. Qt 4.8. Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2012. – 912 с.

18) Документация к библиотеке OpenCV [Электронный ресурс] URL: http://opencv.org/documentation.html, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 27.05.2016 г.

19) Страуструп Б. Язык программирования C++. М.: Бином, 2011. – 1136 с.

20) ГОСТ Р ИСО 26000-2012 Руководство по социальной ответственности. М.: Стандартинформ, 2014. – 114 с.

21) Макконелл С. Совершенный код. Мастер-класс / Пер. с англ. – М. : Издательсство «Русская редакция», 2014. – 896 стр.

22) ГОСТ 13109-97 Электрическая энергия. Совместимость технических средств электромагнитная. Нормы качества электрической энергии в системах электроснабжения общего назначения. М. : Стандартинформ 2006. – 31 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг программы

В настоящем приложении представлено описание классов приложения для оценки отклонения результатов ручной и автоматической сегментации цифровых изображений.

А.1 Описание класса EGBS

```
class EGBS {
 public:
     EGBS();
     ~EGBS();

     int applySegmentation( Mat& image, float sigma, float threshold, int
min_component_size );
     Mat recolor( bool random_color = false );
     int noOfConnectedComponents();

 protected:
     Mat image;
     Size imageSize;
     DisjointSetForest forest;
     inline float diff( Mat& rgb, int x1, int y1, int x2, int y2 );
       };


/**
  * Вычисляем разность между 2 пикселями (3 канала)
  * взяв евклидову норму этой разности.
  */
 float EGBS::diff( Mat& rgb, int x1, int y1, int x2, int y2 ) {
     Vec3f pix1 = rgb.at<Vec3f>(y1, x1);
     Vec3f pix2 = rgb.at<Vec3f>(y2, x2);
     return sqrt( (pix1 - pix2).dot((pix1 - pix2)) );
        }


/**
  * Выполнение сегментации
  */
 int EGBS::applySegmentation( Mat& image, float sigma, float threshold, int
min_component_size ) {
     this->image = image.clone();
     this->imageSize = image.size();

     /* Размытие по Гаусу */
     Mat smoothed;
     image.convertTo( smoothed, CV_32FC1 );
     GaussianBlur( smoothed, smoothed, Size(5,5), sigma );

     vector<Edge> edges( imageSize.area() * 4 );
     int no_of_edges = 0;
```

```
    /* Создание ребер между каждыми пикселями , с весом */
    for( int y = 0; y < imageSize.height; y++ ) {
        for( int x = 0; x < imageSize.width; x++ ) {
            if( x < imageSize.width - 1 ){
                edges[no_of_edges].a        = y * imageSize.width + x;
                edges[no_of_edges].b        = y * imageSize.width + (x + 1);
                edges[no_of_edges].weight   = diff( smoothed, x, y, x + 1, y );
                no_of_edges++;
            }
            if( y < imageSize.height - 1 ) {
                edges[no_of_edges].a        = y       * imageSize.width + x;
                edges[no_of_edges].b        = (y + 1) * imageSize.width + x;
                edges[no_of_edges].weight   = diff( smoothed, x, y, x, y + 1 );
                no_of_edges++;
            }
            if( (x < imageSize.width - 1) && (y < imageSize.height - 1) ) {
                edges[no_of_edges].a        = y       * imageSize.width + x;
             edges[no_of_edges].b        = (y + 1) * imageSize.width + (x + 1);
            edges[no_of_edges].weight   = diff( smoothed, x, y, x + 1, y + 1 );
                no_of_edges++;
            }
            if( (x < imageSize.width - 1) && (y > 0) ) {
                edges[no_of_edges].a        = y       * imageSize.width + x;
             edges[no_of_edges].b        = (y - 1) * imageSize.width + (x + 1);
            edges[no_of_edges].weight   = diff( smoothed, x, y, x + 1, y - 1 );
                no_of_edges++;
            }
        }
    }
    edges.resize( no_of_edges );
    /* выполнить сегментацию */
    forest.segmentGraph( imageSize.height * imageSize.width, no_of_edges,
edges, threshold );
    /* Объединяем все маленькие элементы */
    for( Edge& edge: edges ) {
        int a = forest.find( edge.a );
        int b = forest.find( edge.b );
        if( (a != b) && (( forest.size(a) < min_component_size) ||
(forest.size(b) < min_component_size)) ) {
            forest.join( a, b );
        }
    }
    return forest.noOfElements();
      }
```

A.2 Описание класса DisjointSetForest

```
// Система непересекающихся множеств
 struct SetNode {
    int rank;
    int parent;
    int size;
 };
// Взвешенные ребра

 struct Edge {
    int a;
    int b;
    float weight;
```

```
    bool operator<( const Edge& other ) {
        return weight < other.weight;
    }
};
class DisjointSetForest{
public:
    DisjointSetForest();
    DisjointSetForest( int no_of_elements );
    ~DisjointSetForest();

    void init( int no_of_elements );
    int find( int x );
    void join( int x, int y );
    int size( int x );
    int noOfElements();

    void segmentGraph( int no_of_vertices, int no_of_edges, vector<Edge>&
edges, float c );

private:
    vector<SetNode> elements;
    int num;
      };


// найти элемент
int DisjointSetForest::find( int x ) {
    int y = x;
    while( y != elements[y].parent )
        y = elements[y].parent;
    elements[x].parent = y;
    return y;
}
// объединить наборы
void DisjointSetForest::join( int x, int y ) {
    if ( elements[x].rank > elements[y].rank ) {
        elements[y].parent  = x;
        elements[x].size    += elements[y].size;
    }
    else {
        elements[x].parent  = y;
        elements[y].size    += elements[x].size;

        if( elements[x].rank == elements[y].rank )
            elements[y].rank++;
    }
    num--;
}
// размер набора
int DisjointSetForest::size( int x ) {
    return elements[x].size;
}
// общее колличество наборов внутри леса
int DisjointSetForest::noOfElements() {
    return num;
}
// сегментирование графа с учетом весов
void DisjointSetForest::segmentGraph( int no_of_vertices, int no_of_edges,
vector<Edge>& edges, float c ) {
    init( no_of_vertices );

    sort( edges.begin(), edges.end(), []( Edge& a, Edge& b){
        return a.weight < b.weight;
```

```
    });

    vector<float> thresholds( no_of_vertices, c );


    for( Edge& edge: edges ){
        int a = this->find( edge.a );
        int b = this->find( edge.b );

        if( a != b ) {


            if( edge.weight <= thresholds[a] && edge.weight <= thresholds[b] ) {
                this->join( a, b );
                a = this->find( a );
                thresholds[a] = edge.weight + c / this->size( a );
            }
        }
    }
       }
```

A.3 Описание класса KMeans

```
class KMeans
{
public:
    KMeans();

    cv::Mat applySegmentation(cv::Mat &image, int k);
private:
    int NUM_CLUSTERS;
    cv::Mat image;
    cv::Size imageSize;
    std::vector<Cluster> clusters;
    void clearClusters();
    std::vector<cv::Vec3f> getCentroids();
    void assignCluster(cv::Mat& image);
    inline float diff(cv::Mat& rgb, int x1, int y1, cv::Vec3f ClusterColor);
    void calculateCentroids(cv::Mat& image);
       };


cv::Mat KMeans::applySegmentation(cv::Mat& image, int k)
{
    this->image = image.clone();
    this->imageSize = image.size();
    NUM_CLUSTERS=k;
    bool finish = false;
    int iter;
    cv::Mat img2;
    image.convertTo( img2, CV_32FC1 );

    std::srand(std::time(NULL));

    for (int i=0; i<NUM_CLUSTERS; i++)
    {
        Cluster *cluster = new Cluster(i);
        cv::Vec3f centroid;
        centroid[0]=rand()%255;
        centroid[1]=rand()%255;
        centroid[2]=rand()%255;
```

```
        cluster->setCentroid(centroid);
        clusters.push_back(*cluster);
        delete cluster;
    }

    // начало EM алгоритма
    while (!finish) {
        int exit =0;
        clearClusters();
        std::vector<cv::Vec3f> lastCentroids = getCentroids();
        assignCluster(img2);
        calculateCentroids(img2);
        std::vector<cv::Vec3f> currentCentroids = getCentroids();
        for(int i = 0; i < lastCentroids.size(); i++) {
            cv::Vec3f point1 = lastCentroids[i];
            cv::Vec3f point2 = currentCentroids[i];
            if (((int)point1[0] == (int)point2[0]) &&
((int)point1[1]==(int)point2[1]) && ((int)point1[2]==(int)point2[2]))
            {
                exit++;
            }
        }
        iter++;

        if (exit==k || iter >100) finish = true;

    }


    cv::Mat result( imageSize, CV_8UC3, cv::Scalar(0, 0, 0) );

    for (std::vector<Cluster>::iterator it = clusters.begin() ; it !=
clusters.end(); ++it)
    {
        Cluster cluster = *it;
        std::vector<cv::Point> points = cluster.getPoint();
        cv::Vec3d centroid = cluster.getCentroid();


        for (std::vector<cv::Point>::iterator it2 = points.begin(); it2 !=
points.end(); ++it2)
        {
            cv::Point point = *it2;
            cv::Vec3b centroidNew = centroid;

            result.at<cv::Vec3b>(point)[0]= centroidNew[0];
            result.at<cv::Vec3b>(point)[1]= centroidNew[1];
            result.at<cv::Vec3b>(point)[2]= centroidNew[2];
        }
    }

    return result;

}
```

A.4 Описание класса GMM

```
class GMM
{
public:
    GMM();
    cv::Mat applySegmentation(cv::Mat &image, int k);
private:
    int NUM_CLUSTERS;
```

```
    cv::Mat image;
    cv::Size imageSize;
    std::vector<Cluster> clusters;
    void clearClusters();
    std::vector<cv::Vec3f> getCentroids();
    void assignCluster(cv::Mat& image);
    inline float diff(cv::Mat& rgb, int x1, int y1, cv::Vec3f ClusterColor);
    void calculateCentroids(cv::Mat& image);
    int size;
    long double *wight;
    long double **covariance;
    std::vector<cv::Vec3f> mean;
    long double GaussianDistribution(cv::Vec3f pix, int k);
       };


cv::Mat GMM::applySegmentation(cv::Mat& image, int k)
{
    this->image = image.clone();
    this->imageSize = image.size();
    NUM_CLUSTERS=k;
    bool finish = false;
    int iter;
    cv::Mat img2;
    image.convertTo( img2, CV_32FC1 );

    std::srand(std::time(NULL));

    for (int i=0; i<NUM_CLUSTERS; i++)
    {
        Cluster *cluster = new Cluster(i);
        cv::Vec3f centroid;
        centroid[0]=rand()%255;
        centroid[1]=rand()%255;
        centroid[2]=rand()%255;
        cluster->setCentroid(centroid);
        clusters.push_back(*cluster);
        delete cluster;

    }
    //запуск алгоритма к-средних
    while (!finish) {
        int exit =0;
        clearClusters();
        std::vector<cv::Vec3f> lastCentroids = getCentroids();
        assignCluster(img2);
        calculateCentroids(img2);
        std::vector<cv::Vec3f> currentCentroids = getCentroids();
        //    int distance = 0;
        for(int i = 0; i < lastCentroids.size(); i++) {
            cv::Vec3f point1 = lastCentroids[i];
            cv::Vec3f point2 = currentCentroids[i];
            if (((int)point1[0] == (int)point2[0]) &&
((int)point1[1]==(int)point2[1]) && ((int)point1[2]==(int)point2[2]))
            {
                exit++;
            }
        }
        iter++;

        if (exit==k || iter >100) finish = true;

    }
    //вычислим коэффициенты смешивания (веса) и дисперсию
```

```
    size =imageSize.height*imageSize.width;
    wight = new long double [NUM_CLUSTERS];
    int cluster_i=0;
    covariance = new long double *[NUM_CLUSTERS];
    for (std::vector<Cluster>::iterator it = clusters.begin() ; it !=
clusters.end(); ++it)
    {

        Cluster cluster = *it;
        std::vector<cv::Point> points = cluster.getPoint();
        cv::Vec3d centroid = cluster.getCentroid();
        wight[cluster_i]= points.size() /(double)this->size;
        covariance[cluster_i] = new long double[3];
        double B=0;
        double G=0;
        double R=0;
        for (std::vector<cv::Point>::iterator it2 = points.begin(); it2 !=
points.end(); ++it2)
        {
            cv::Point point = *it2;

            B += pow(img2.at<cv::Vec3f>(point)[0] - centroid[0],2);
            G += pow(img2.at<cv::Vec3f>(point)[1] - centroid[1],2);
            R += pow(img2.at<cv::Vec3f>(point)[2] - centroid[2],2);

        }
        covariance[cluster_i][0] = B/ (long double)this->size;
        covariance[cluster_i][1] = G/ (long double)this->size;
        covariance[cluster_i][2] = R/ (long double)this->size;
        cluster_i++;
    }
    mean = getCentroids();

    //EM alg

    iter=0;
    long double **P = new long double *[size];
    for (int i=0; i<size; i++)
        P[i] = new long double [NUM_CLUSTERS];

    finish = false;

    long double delta;


    //начало EM алгоритма

    while (!finish) {

        int count=0;
        long double likelihoodOld=0;
        long double likelihoodNew=0;

        //E
        for( int y = 0; y < imageSize.height; y++ ) {
            for( int x = 0; x < imageSize.width; x++ ) {
                for (int j=0; j<NUM_CLUSTERS; j++)
                {
                    cv::Vec3f pix = img2.at<cv::Vec3f>(y, x);
                    long double numerator =
wight[j]*GaussianDistribution(pix,j);
                    long double denominator=0;
                    for (int i=0; i<NUM_CLUSTERS; i++)
                    {
```

```
                denominator+=wight[i]*GaussianDistribution(pix,i);
            }

            P[count][j] = numerator/denominator;


            likelihoodOld+=log10(denominator);



        }
        count++;
    }
}

//M

for(int i = 0; i < mean.size(); i++) {
    mean[i][0]=0;
    mean[i][1]=0;
    mean[i][2]=0;
}

long double *Nk = new long double[NUM_CLUSTERS];
for (int j=0; j<NUM_CLUSTERS; j++)
{
    Nk[j]=0;
}
count =0;

for (int j=0; j<NUM_CLUSTERS; j++)
{
    for( int y = 0; y < imageSize.height; y++ ) {
        for( int x = 0; x < imageSize.width; x++ ) {
            Nk[j]+=P[count][j];
            count++;
        }
    }
    count=0;

}


long double *responsibilitie= new long double[3];

// новые центры кластеров
for (int j=0; j<NUM_CLUSTERS; j++)
{

    for (int i=0; i<3; i++)
    {
        responsibilitie[i]=0;
    }
    int count2=0;
    for( int y = 0; y < imageSize.height; y++ ) {
        for( int x = 0; x < imageSize.width; x++ ) {

            cv::Vec3f pix = img2.at<cv::Vec3f>(y, x);
            responsibilitie[0]+= pix[0]*P[count2][j];
            responsibilitie[1]+= pix[1]*P[count2][j];
            responsibilitie[2]+= pix[2]*P[count2][j];
            count2++;
        }
    }
```

```
    mean[j][0]=responsibilitie[0]/Nk[j];
    mean[j][1]=responsibilitie[1]/Nk[j];
    mean[j][2]=responsibilitie[2]/Nk[j];

}
//новые матрицы ковариаций
for (int j=0; j<NUM_CLUSTERS; j++)
{

    for (int i=0; i<3; i++)
    {
        responsibilitie[i]=0;
    }
    int count3=0;
    for( int y = 0; y < imageSize.height; y++ ) {
        for( int x = 0; x < imageSize.width; x++ ) {
            cv::Vec3f pix = img2.at<cv::Vec3f>(y, x);
            long double buf1 = (pix[0]-mean[j][0])*(pix[0]-mean[j][0]);
            long double buf2 = (pix[1]-mean[j][1])*(pix[1]-mean[j][1]);
            long double buf3 = (pix[2]-mean[j][2])*(pix[2]-mean[j][2]);

            responsibilitie[0]+= buf1*P[count3][j];
            responsibilitie[1]+= buf2*P[count3][j];
            responsibilitie[2]+= buf3*P[count3][j];
            count3++;
        }
    }
    covariance[j][0]=responsibilitie[0]/Nk[j];
    covariance[j][1]=responsibilitie[1]/Nk[j];
    covariance[j][2]=responsibilitie[2]/Nk[j];

}

for (int j=0; j<NUM_CLUSTERS; j++) {
    for (int z=0; z<3; z++)
    {
        if (covariance[j][z]==0) {
            covariance[j][z]=rand()%255;
            mean[j][z]=rand()%255;
        }
    }
}
delete [] responsibilitie;

for (int j=0; j<NUM_CLUSTERS; j++)
{
    wight[j]=Nk[j]/(double)this->size;
}

delete [] Nk;

//логарифм новой функции правдоподобия
for( int y = 0; y < imageSize.height; y++ ) {
    for( int x = 0; x < imageSize.width; x++ ) {
        for (int j=0; j<NUM_CLUSTERS; j++)
        {
            cv::Vec3f pix = img2.at<cv::Vec3f>(y, x);

            long double denominator=0;
            for (int i=0; i<NUM_CLUSTERS; i++)
            {
                denominator+=wight[i]*GaussianDistribution(pix,i);
            }
```

```
                    likelihoodNew+=log10(denominator);


                }

            }
        }
        delta = (likelihoodNew-likelihoodOld)/(long
double)std::abs(likelihoodOld);

        iter++;

        if (iter >100 || delta < 5e-4) finish = true;
      }

    clearClusters();
    int count=0;


    for( int y = 0; y < imageSize.height; y++ ) {
        for( int x = 0; x < imageSize.width; x++ ) {
            double distance = 0.0;
            int cluster = 0;
            for (int j=0; j<NUM_CLUSTERS; j++)
            {
                distance = P[count][j];
                if (distance>0.5)
                {
                    cv::Point point;
                    point.x=x;
                    point.y=y;
                    clusters[j].addPoint(point);

                }

            }
            count++;
        }
    }


    for (std::vector<Cluster>::iterator it = clusters.begin() ; it !=
clusters.end(); ++it)
    {
        delete covariance[cluster_i];

    }
    delete [] covariance;

    delete [] wight;

    for (int i=0; i<size; i++)
        delete []P[i];
    delete []P;


    cv::Mat result( imageSize, CV_8UC3, cv::Scalar(0, 0, 0) );

    for (std::vector<Cluster>::iterator it = clusters.begin() ; it !=
clusters.end(); ++it)
    {
        Cluster cluster = *it;
```

```
        std::vector<cv::Point> points = cluster.getPoint();
        cv::Vec3d centroid = cluster.getCentroid();
        for (std::vector<cv::Point>::iterator it2 = points.begin(); it2 !=
points.end(); ++it2)
        {
            cv::Point point = *it2;
            cv::Vec3b centroidNew = centroid;

            result.at<cv::Vec3b>(point)[0]= centroidNew[0];
            result.at<cv::Vec3b>(point)[1]= centroidNew[1];
            result.at<cv::Vec3b>(point)[2]= centroidNew[2];
        }
    }
    clusters.clear();
    std::vector<Cluster>().swap(clusters);
    mean.clear();
    std::vector<cv::Vec3f>().swap(mean);


    return result;

       }
```

A.5 Описание класса Cluster

```
class Cluster
{
public:
    Cluster(int id)
    {
        this->id=id;
    }

    Cluster();
    ~Cluster() {}

    std::vector<cv::Point> getPoint()
    {
        return points;
    }
    void addPoint(cv::Point point)
    {
        points.push_back(point);
    }
    void setPoint(std::vector<cv::Point> points)
    {
        this->points=points;
    }
    cv::Vec3f getCentroid()
    {
        return this->centroid;
    }
    void setCentroid(cv::Vec3f centroid)
    {
        this->centroid=centroid;
    }
    int getId()
    {
        return this->id;
    }
    void clear()
```

```
    {
        points.clear();
    }

private:
    std::vector<cv::Point> points;
    cv::Vec3f centroid;
    int id;

       };
```

A.5 Описание класса PixmapWidget

```
class PixmapWidget : public QLabel
{
    Q_OBJECT

public :
    PixmapWidget( QWidget* parent = 0 );
    void set_qimage0(const QImage& qimage1);
    void set_qimage(const QImage& qimage1);
    void set_doWheelEvent(bool doWheelEvent1);
    void set_zoomFactor(float zoomFactor1);
    float get_zoomFactor() const;
    int getPixWidth() const;
    int getPixHeight() const;
    int get_xoffset() const;
    int get_yoffset() const;
    void set_hasText(bool hasText1);
    const QImage& get_qimage() const;
    void set_text(QString text1);

public slots :
    void setZoomFactor(float f);

signals :
    void zoomFactorChanged(float f);

protected :
    void paintEvent(QPaintEvent* event);

private :
    QImage qimage;
    float zoomFactor;
    int xoffset;
    int yoffset;
    bool hasText;
    QString text;
    bool doWheelEvent;
       };
```

A.6 Описание класса MainWindow

```
class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = 0);
```

```
    ~MainWindow();

private slots:

    void openFileName();
    void openFileNameA();
    void openFileNameB();
    void deleteList();
    void openRecentFile();
    void saveImage();
    void zoomIn();
    void zoomOut();
    void normalSize();
    void start();
    void evaluate();
    void do_scale0(int value);
    void settings();
    void evaluation();
    void default_settings();
    void scaleA(int value);
    void scaleB(int value);
private:
    PixmapWidget* imageLabel;
    QLabel* iL;
    QScrollArea* scrollArea;
    QSpinBox* scale_spin0;
    QSlider* scale_slider0;
    QLabel* Sigma_text;
    QLabel* Min_text;
    QLabel* threshold_text;
    QLabel* MeanNumberK_text;
    QLabel* MeanNumberGMM_text;
    QStackedWidget* stackedWidget;
    // основные функции
    void open();
    void createActions();
    void createMenus();
    void scaleImage(double factor);
    double scaleFactor;
    bool eventFilter(QObject* object, QEvent* event);
    QString fileName;
    int img_height;
    int img_width;
    int img_size;
    QImage img;
    QImage image_result;
    cv::Mat mat;
    EGBS egbs;
    QStringList nameFilters;
    QString last_directory_used;
    void updateRecentFileActions();
    QString strippedName(const QString &fullFileName);
    void setCurrentFile(const QString &fileName1);
    QString curFile;
    QDialog* settings_window;
    QDialogButtonBox* dial_buttons;
    QVBoxLayout* algorithm();
    QRadioButton* egbs_radio;
    QRadioButton* kmean_radio;
    QRadioButton* gmm_radio;
    QDoubleSpinBox* sigma_spin;
    QSpinBox* threshold_spin;
    QSpinBox* min_component_size_spin;
    QSpinBox* kmean_spin;
```

```
    QSpinBox* gmm_spin;
    float sigma;
    float threshold;
    int min_component_size;
    int mean_k;
    int mean_gmm;
    int model;
    // Ok
    void apply_settings();
    // отмена
    void cancel_settings();
    //отклонение
    QDialog* evaluation_window;
    void open_imageA();
    void open_imageB();
    QString fileNameA;
    QString fileNameB;
    QScrollArea* scrollAreaA;
    PixmapWidget* imageLabelA;
    QScrollArea* scrollAreaB;
    PixmapWidget* imageLabelB;
    QSpinBox* scale_spinA;
    QSpinBox* scale_spinB;
    QSpinBox *sample_spin;
    QLabel* label_name1;
    QLabel* label_name2;
    QLabel* text_A;
    QLabel* text_B;
    QLabel* eval;
    QImage image1;
    QImage image2;
    int imgA_width;
    int imgA_height;
    int imgA_size;
    int imgB_width;
    int imgB_height;
    int imgB_size;
    QRadioButton *egbs_radio_button;
    QRadioButton *kmean_radio_button;
    QRadioButton *gmm_radio_button;
    int model_evaluation;
protected:
    void dragEnterEvent(QDragEnterEvent* event);
    void dragMoveEvent(QDragMoveEvent* event);
    void dropEvent(QDropEvent* event);
    void dragLeaveEvent(QDragLeaveEvent* event);

    void dragEnterEventA(QDragEnterEvent* event);
    void dragMoveEventA(QDragMoveEvent* event);
    void dropEventA(QDropEvent* event);
    void dragLeaveEventA(QDragLeaveEvent* event);

    void dragEnterEventB(QDragEnterEvent* event);
    void dragMoveEventB(QDragMoveEvent* event);
    void dropEventB(QDropEvent* event);
    void dragLeaveEventB(QDragLeaveEvent* event);

    QAction* openAct;
    QAction* separatorAct;
    enum { MaxRecentFiles = 5 };
    QAction* recentFileActs[MaxRecentFiles];
    QAction* deleteAct;
    QAction* saveAct;
    QAction* exitAct;
```

```
    QAction* zoomInAct;
    QAction* zoomOutAct;
    QAction* normalSizeAct;
    QAction* startAct;
    QAction* settingsAct;
    QAction* evaluateAct;

    QMenu* fileMenu;
    QMenu* viewMenu;
    QMenu* segmentationMenu;

signals :
    void changed(const QMimeData* mimeData = 0);
       };


void MainWindow::start()
{
    if(model==1){

    mat= cv::imread(fileName.toLatin1().data());
    egbs.applySegmentation( mat, sigma, threshold, min_component_size );
    cv::Mat random_color  = egbs.recolor(false);
    cv::cvtColor(random_color,random_color,CV_BGR2RGB);
    image_result= QImage((const unsigned char*)(random_color.data),
random_color.cols,random_color.rows,QImage::Format_RGB888);
    image_result.bits();
    QApplication::processEvents();
    imageLabel->set_qimage(image_result);

    }

    if (model==2) {
        mat= cv::imread(fileName.toLatin1().data());
        KMeans kmeans;
        cv::Mat k = kmeans.applySegmentation( mat, mean_k);
        cv::cvtColor(k,k,CV_BGR2RGB);
        image_result= QImage((const unsigned char*)(k.data),
k.cols,k.rows,QImage::Format_RGB888);
        image_result.bits();
        QApplication::processEvents();
        imageLabel->set_qimage(image_result);

    }


    if (model==3) {
        mat= cv::imread(fileName.toLatin1().data());
        GMM gmm;
        cv::Mat g = gmm.applySegmentation( mat, mean_gmm);
        cv::cvtColor(g,g,CV_BGR2RGB);
        image_result= QImage((const unsigned char*)(g.data),
g.cols,g.rows,QImage::Format_RGB888);
        image_result.bits();
        QApplication::processEvents();
        imageLabel->set_qimage(image_result);

    }

    if( model == 1 )
    {
        stackedWidget->setCurrentIndex(0);
    }
```

```
    if( model == 2 )
    {
        stackedWidget->setCurrentIndex(1);
    }
    if( model == 3 )
    {
        stackedWidget->setCurrentIndex(2);
    }
       }


void MainWindow::evaluate()
{
    int sample = sample_spin->value();
    float zxc=0;
    float max=0;

    if (egbs_radio_button->isChecked())
    {
        for (int g=0; g<sample;g++)
        {
            cv::Mat matA= cv::imread(fileNameA.toLatin1().data());
            cv::Mat matB= cv::imread(fileNameB.toLatin1().data());

            float r = static_cast <float> (rand()) / static_cast <float>
(RAND_MAX);
            float th = rand()%5000;
            int mcs = rand()%500;

            egbs.applySegmentation( matA, r, th, mcs);
            cv::Mat random_color  = egbs.recolor(false);


            cv::Mat gray, src_gray, edge, draw;
            cv::cvtColor(random_color, src_gray, CV_BGR2GRAY);
            cv::blur( src_gray, gray, cv::Size(3,3) );
            cv::Canny( gray, edge, 50, 150, 3, true );
            cv::Mat contoursInv;
            cv::threshold(edge,contoursInv,128,255,  cv::THRESH_BINARY_INV);
            contoursInv.convertTo(draw, CV_8U);
            cv::Size imageSize = matA.size();
            for( int y = 0; y < imageSize.height; y++ ) {
                for( int x = 0; x < imageSize.width; x++ ) {

                    if (draw.at<uchar>(y,x)==0)
                    {
                        matA.at<cv::Vec3b>(y, x)[0]=0;
                        matA.at<cv::Vec3b>(y, x)[1]=0;
                        matA.at<cv::Vec3b>(y, x)[2]=254;

                    }

                }
            }
            int v=0;
            int z=0;


            for( int y = 0; y < imageSize.height; y++ ) {
                for( int x = 0; x < imageSize.width; x++ ) {
                    cv::Vec3b pixA = matB.at<cv::Vec3b>(y, x);
                    if (pixA[0]==0 and pixA[1]==0 and pixA[2]==254)
                    {
                        cv::Vec3b pixB = matA.at<cv::Vec3b>(y, x);
```

```
                        if (pixB[0]==0 and pixB[1]==0 and pixB[2]==254)
                        {
                            v++;
                        }

                        z++;

                    }

                }
            }

            zxc=(v*100)/(double)z;
            if (zxc>max)
            {
                max=zxc;
                cv::cvtColor(matA,matA,CV_BGR2RGB);
                image_result= QImage((const unsigned char*)(matA.data),
matA.cols,matA.rows,QImage::Format_RGB888);
                image_result.bits();
                QApplication::processEvents();
                imageLabelA->set_qimage(image_result);

            }
        }

        eval->setText(tr("Отклонение = ")+QString::number(100-max));

    }
    if (kmean_radio_button->isChecked())
    {
        for (int g=0; g<sample;g++)
        {
            cv::Mat matA= cv::imread(fileNameA.toLatin1().data());
            cv::Mat matB= cv::imread(fileNameB.toLatin1().data());
            int mn = rand()%6+1;
            KMeans kmeans;
            cv::Mat k = kmeans.applySegmentation( matA, mn);



            cv::Mat gray, src_gray, edge, draw;
            cv::cvtColor(k, src_gray, CV_BGR2GRAY);
            cv::blur( src_gray, gray, cv::Size(3,3) );
            cv::Canny( gray, edge, 50, 150, 3, true );
            cv::Mat contoursInv;
            cv::threshold(edge,contoursInv,128,255,  cv::THRESH_BINARY_INV);


            contoursInv.convertTo(draw, CV_8U);

            cv::Size imageSize = matA.size();

            for( int y = 0; y < imageSize.height; y++ ) {
                for( int x = 0; x < imageSize.width; x++ ) {

                    if (draw.at<uchar>(y,x)==0)
                    {
                        matA.at<cv::Vec3b>(y, x)[0]=0;
                        matA.at<cv::Vec3b>(y, x)[1]=0;
                        matA.at<cv::Vec3b>(y, x)[2]=254;

                    }
```

```
            }
        }



        int v=0;
        int z=0;


        for( int y = 0; y < imageSize.height; y++ ) {
            for( int x = 0; x < imageSize.width; x++ ) {
                cv::Vec3b pixA = matB.at<cv::Vec3b>(y, x);
                if (pixA[0]==0 and pixA[1]==0 and pixA[2]==254)
                {
                    cv::Vec3b pixB = matA.at<cv::Vec3b>(y, x);
                    if (pixB[0]==0 and pixB[1]==0 and pixB[2]==254)
                    {
                        v++;
                    }

                    z++;

                }


            }
        }



        zxc=(v*100)/(double)z;
        if (zxc>max)
        {
            max=zxc;
            cv::cvtColor(matA,matA,CV_BGR2RGB);
            image_result= QImage((const unsigned char*)(matA.data),
matA.cols,matA.rows,QImage::Format_RGB888);
            image_result.bits();
            QApplication::processEvents();
            imageLabelA->set_qimage(image_result);

        }



    }
    eval->setText(tr("Отклонение = ")+QString::number(100-max));

}
if (gmm_radio_button->isChecked())
{
    for (int g=0; g<sample;g++)
    {
        cv::Mat matA= cv::imread(fileNameA.toLatin1().data());
        cv::Mat matB= cv::imread(fileNameB.toLatin1().data());
        int mn = rand()%3+2;

        GMM GMM;
        cv::Mat gmm = GMM.applySegmentation( matA, mn);

        cv::Mat gray,src_gray, edge, draw;
```

```
            cv::cvtColor(gmm, src_gray, CV_BGR2GRAY);
            cv::blur( src_gray, gray, cv::Size(3,3) );
            cv::Canny( gray, edge, 50, 150, 3, true );
            cv::Mat contoursInv;
            cv::threshold(edge,contoursInv,128,255,  cv::THRESH_BINARY_INV);
            contoursInv.convertTo(draw, CV_8U);
            cv::Size imageSize = matA.size();
            for( int y = 0; y < imageSize.height; y++ ) {
                for( int x = 0; x < imageSize.width; x++ ) {

                    if (draw.at<uchar>(y,x)==0)
                    {
                        matA.at<cv::Vec3b>(y, x)[0]=0;
                        matA.at<cv::Vec3b>(y, x)[1]=0;
                        matA.at<cv::Vec3b>(y, x)[2]=254;

                    }


                }
            }


            int v=0;
            int z=0;


            for( int y = 0; y < imageSize.height; y++ ) {
                for( int x = 0; x < imageSize.width; x++ ) {
                    cv::Vec3b pixA = matB.at<cv::Vec3b>(y, x);
                    if (pixA[0]==0 and pixA[1]==0 and pixA[2]==254)
                    {
                        cv::Vec3b pixB = matA.at<cv::Vec3b>(y, x);
                        if (pixB[0]==0 and pixB[1]==0 and pixB[2]==254)
                        {
                            v++;
                        }

                        z++;

                    }


                }
            }



            zxc=(v*100)/(double)z;
            if (zxc>max)
            {
                max=zxc;
                cv::cvtColor(matA,matA,CV_BGR2RGB);
                image_result= QImage((const unsigned char*)(matA.data),
matA.cols,matA.rows,QImage::Format_RGB888);
                image_result.bits();
                QApplication::processEvents();
                imageLabelA->set_qimage(image_result);

            }

        }
        eval->setText(tr("Отклонение = ")+QString::number(100-max));
```

```
    }


        }
```