

Содержание

ВВЕДЕНИЕ	6
1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧ	8
1.1. Актуальность разрабатываемого приложения.....	8
1.2. Постановка задачи	12
1.3. Определение предметной области дипломного проекта.....	12
1.3.1. Исходные данные: изображение.....	12
1.3.2. Фильтрация.....	13
1.3.3. Обработка изображения детектором границ Кенни (Canny)	14
1.3.4. Обработка изображений преобразованием Хафа.....	15
1.3.5. Бинаризация изображения	17
1.3.6. Алгоритм кластеризации K-Means	17
1.3.7. Градиентный спуск и его применение в нейронных сетях	18
1.3.8. Нейронные сети и Метод обратного распространения ошибки	19
1.3.9. Сохранение и обработка данных	20
2. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	21
2.1. Алгоритмы предобработки и отчистки изображения	21
2.1.1. Уменьшение размерности входных данных.....	21
2.1.2. Очистка изображения	25
2.1.3. Устранения наклона изображения.....	27
2.1.4. Обнаружение таблицы.....	33
2.2. Распознавание текстов	35
2.2.1. Алгоритм метода обратного распространения ошибки	36
2.2.2. Распознавание букв	38
2.3. Исключение ключевых данных	39
2.3.1. Расстояние Левенштейна и нечёткий поиск строки	39
2.3.2. Обнаружение паттерна документа и исключение ключевых слов	41
3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ.....	42
3.1. Описание используемых инструментов и технологий	42
3.1.1. Библиотека OpenCV	42
3.1.2. Библиотека Tesseract.....	46
3.2. Структура базы данных	47
3.2.1. Диаграмма сущности-связи	48
3.2.2. Объектно-реляционное отображение с помощью Hibernate.....	50
3.3. Описание основных классов и методов модуля.....	51
3.3.1. Модуль TabulatedOCR	51

Иzm.	Лист	№ докум.	Подпись	Дат	СОДЕРЖАНИЕ		
Разраб.	Манитариву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей		
Руковод.	Брусенцев А.И.						
Консул.							
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.				Лит. Лист Листов		
					3 116		
					БГТУ им. В. Г. Шухова, ПВ-51		

3.3.2.	Модуль DBManager	53
3.3.3.	Модуль ImageProcessing.....	54
3.3.4.	Модуль TableStructureDetection.....	55
3.3.5.	Модуль TextExtraction	59
4.	ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ	61
4.1.	Вычислительный эксперимент 1: форматированный текстовой документ	61
4.1.1.	Входные данные.....	61
4.1.2.	Выходные данные.....	63
4.1.3.	Анализ результата.....	63
4.2.	Вычислительный эксперимент 2: документ с таблицей определённой структуры	64
4.2.1.	Входные данные.....	64
4.2.2.	Выходные данные.....	65
4.2.3.	Анализ результата.....	68
	Вывод	68
5.	РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ.....	69
5.1.	ВВЕДЕНИЕ	69
5.2.	НАСТРОЙКА.....	70
5.3.	ПРОЕКТ.....	71
5.4.	ШАБЛОНЫ	72
5.4.1.	Табличный формат	72
5.4.2.	Текстовой формат	73
5.5.	ЗАПУСК.....	74
5.6.	ПРОСМОТР ПРОМЕЖУТОЧНЫХ РЕЗУЛЬТАТОВ	74
6.	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА.....	76
6.1.	НАЗНАЧЕНИЕ ПРОГРАММНОГО ПРОДУКТА	76
6.2.	РАСЧЕТ ЕДИНОВРЕМЕННЫХ ЗАТРАТ НА РАЗРАБОТКУ ПО.....	76
6.2.1.	Материальные затраты	77
6.2.2.	Основная и дополнительная заработка платы.....	78
6.2.3.	Отчисления на социальные нужды	79
6.2.4.	Стоимость машинного времени.....	79
6.2.5.	Стоимость инструментальных средств.....	80
6.2.6.	Затраты на интернет	80
6.2.7.	Накладные расходы	80
6.2.8.	Смета затрат на разработку ПО	81
6.3.	ИНВЕСТИЦИОННЫЙ ПЛАН.....	81
6.4.	ГРАФИК РЕАЛИЗАЦИИ ПРОЕКТА.....	82
6.5.	АНАЛИЗ РЫНОЧНЫХ ВОЗМОЖНОСТЕЙ ПРОГРАММНОГО ПРОДУКТА.....	83
6.6.	ПЛАН ТИРАЖИРОВАНИЯ И РЕАЛИЗАЦИИ ПРОГРАММНОГО ПРОДУКТА	84
6.7.	СМЕТА ЗАТРАТ НА ТИРАЖИРОВАНИЕ И РЕКЛАМУ	85
6.8.	ПЛАН ПРИБЫЛИ ОТ ПРОДАЖ	85

Изм.	Лист	№ докум.	Подпись	Дата

СОДЕРЖАНИЕ

Лист
4

6.9.	ФИНАНСОВЫЙ ПЛАН ПРОЕКТА	85
6.10.	ПОКАЗАТЕЛИ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	87
6.10.1.	Интегральный экономический эффект	88
6.10.2.	Индекс доходности (SRR).....	89
6.10.3.	Внутренний коэффициент эффективности проекта (IRR).	89
6.11.	ПРАВОВЫЕ АСПЕКТЫ	91
6.11.1.	Лицензионное соглашение	92
6.11.2.	Авторское право	92
6.11.3.	Ограничения.....	92
6.11.4.	Ответственность Производителя	93
ЗАКЛЮЧЕНИЕ		94
СПИСОК ЛИТЕРАТУРЫ		95
ПРИЛОЖЕНИЕ А.....		98
ПРИЛОЖЕНИЕ Б		104
ПРИЛОЖЕНИЕ В		109
ПРИЛОЖЕНИЕ Г		112

Изм.	Лист	№ докум.	Подпись	Дата

СОДЕРЖАНИЕ

Лист
5

Введение

Наименование: «Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей».

Приложение позволит распознавать форматированный текст и табличную структуру в изображениях, находящихся на локальном диске или снятых с помощью фотоаппарата смартфона, выбирать ключевую информацию из считанных текстов или таблиц, обрабатывать эту информацию и создавать XML-файл или в SQL-скрипт для упрощения дальнейшей работы с этими данными.

Приложение будет доступно на всех операционных системах (Unix, Linux, Windows, MacOS, BSD) поддерживающих Java 7 и более позднюю.

Входные данные являются изображением, представляющим собой скан документа, и описание шаблона этого документа. Программа может распознавать два типа шаблона:

- *форматированный текст*, указывающий констант и переменные. Под константами понимаются те части документа, которые остаются одинаковыми для разного документа одного типа. Под переменными понимаются те части документа, которые отличают один документ от другого, имеющего того же типа. Нахождение значения этих переменных является основной задачей программы при распознавании документа по шаблону форматированного текста.
- *Структура таблицы*, в которой указаны количества и названия столбцы. Обнаружение структуры таблицы и распознавание текстов из каждой клетки является основной задачей программы при распознавании документа по шаблону структуры таблицы.

Изм.	Лист	№ докум.	Подпись	Дат	ВВЕДЕНИЕ		
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей		
Руковод.	Брусенцев А.И.				Lит.	Лист	Листов
Консул.						6	116
Н. контр.	Полунин А.И.				БГТУ им. В. Г. Шухова, ПВ-51		
Зав. каф.	Поляков В.М.						

Для решения задач рассматривается последовательность методов и алгоритмов. Ключевыми алгоритмами являются алгоритм метода Кэнни (для обнаружения границы форм букв и таблицы в изображении), алгоритм преобразования Хафа (для определения наклона и нахождения ребер таблицы) и алгоритм обучения нейронной сети «Метод обратного распространения ошибки» (для распознавания текстов). Также рассматриваются другие алгоритмы и методы для предобработки изображения, в том числе разные методы преобразования цветного изображения в чёрно-белое, алгоритм кластеризации методом K-Means. Также для распознавания и извлечения информации помогают метод скользящего окна, нечёткий поиск строки и расстояние Левенштейна.

Изм.	Лист	№ докум.	Подпись	Дата	ВВЕДЕНИЕ	Лист
						7

1. Описание предметной области и анализ методов решения задач

1.1. Актуальность разрабатываемого приложения

Большая часть профессиональных и даже личных отношений современного человека осуществляется при помощи компьютерной техники. Предприятие любого размера не может выжить в этой информационной эпохе без использования компьютеров. Успешные компании в текущее время зачастую проводят глубокий анализ имеющихся данных для того, чтобы принять решение и сделать прогнозы о развитии компании. Но такого рода анализы можно производить только с цифровыми данными. Современный бизнес работает гораздо быстрее и эффективнее, благодаря оцифровке всех своих данных.

Однако, во многих компаниях, особенно в технологически менее развитых странах, таких как Мадагаскар, до сих пор массово используют физические носители для бизнеса и имеют массивные архивы в виде бумажных документов. Рано или поздно эти предприятия должны перейти от этой привычки 20-ого века в информационную эру 21-ого века и начать оцифровывать все свои документы и массивные архивы. Но ввод каждой записи в базу данных вручную является утомительной работой и занимает длительное время.

Оцифровка данных и архивов поспособствует не только освобождению места и проведению глубокого анализа над историческими данными, но также повышению безопасности хранения информации. Одним из многих случаев, которые показывают ненадежность хранения документов в бумажном виде, является масштабный пожар в библиотеке на Нахимовском проспекте в Москве в конце января этого года (31/01/2015) ^[1]. Фундаментальная библиотека Института Научной Информации по Общественным Наукам РАН,

Изм.	Лист	№ докум.	Подпись	Дат	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ		
					Лит.	Лист	Листов
Разраб.	Манитрапишу А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	8	116
Руковод.	Брусенцев А.И.						
Консул.							
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.						
					БГТУ им. В. Г. Шухова, ПВ-51		

созданная в 1918 году, имеет статус библиотеки федерального значения. В ее фондах хранится 14,2 млн экземпляров документов на древних, современных восточных, европейских и русском языках. Также там хранятся редкие издания XVI — начала XX веков. В этой библиотеке имелись самые полные, а в отдельных случаях и единственныес в России собрания документов Лиги наций и международных организаций, например, ООН, МОТ, ЮНЕСКО [2]. Там находится одна из крупнейших в России коллекций книг на славянских языках. Из-за ненадежности бумажного носителя все эти важнейшие знания потеряны. Эта информация могла бы сохраниться, если бы ее вовремя оцифровали. Но оцифровка 1,4 миллионов документов невозможна без АС, предназначенной для этого.

Помимо оцифровки архивов многие компании в развитых странах также получают от клиентов документы в бумажной форме, такие как выписка банковского счета и банковские транзакции, чеки, паспорта, квитанции, страховые документы и т.п. В отделах приема клиентов эта информация до сих пор вводится вручную. Использование автоматизированной системы ввода данных может сэкономить всего несколько минут на одном клиенте, зато в широком масштабе компания получит огромный выигрыш времени при большом количестве обслуживаемых клиентов.

Для первого конкретного примера можно взять процесс ввода личных данных клиентов в базу данных. Известно, что эти данные всегда читаются из одного и того же документа с одним и тем же форматом: паспорт.

Как показано на шаблоне на рисунке 1.1, очевидно, что этот процесс можно автоматизировать, так как при известном формате исходного документа паспорт или загранпаспорт, расположение интересующей информации известно. Однако, общеизвестно, что во всех пунктах приема клиентов, где нужно обрабатывать паспортные данные (проверка их наличия в базе или создание новой записи в базе), ввод данных без ошибок занимает от

Иzm.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						9

40 секунд до 5 минут. Использование АС для таких задач может существенно сократить потраченное время.

Другой конкретный пример возможного применения программ для автоматизации в компании - ввод данных для бухгалтерского отчёта в одном из филиалов компании WWF (*World Wildlife Fund* - рус. Всемирный фонд природы). WWF – это крупнейшая в мире независимая природоохранная организация с более чем 5 миллионами участников во всём мире, работающая в более чем 100 странах, поддерживающая около 1300 природоохраных проектов по всему миру^[3]. Одним из сотни филиалов WWF Мадагаскара является WWF Fandriana в области Fianarantsoa.



Рисунок 1.1 - Паспорт и заграничный паспорт гражданина РФ содержат одни и те же данные, отличаются они только расположением этих данных, т.е. форматом текста в документе

На рисунке 1.2 показаны два конкретных разных документа о денежной транзакции в этом филиале, информация из которых должна быть введена в базу данных или в электронные архивы бухгалтером. На первом рисунке есть банковская выписка от банка BFV-SG. Из этого документа, например, нужно вводить в бухгалтерский расчёт только информацию о дате всех транзакций, описание транзакции, суммы и номера выписываемого банковского счёта. Второй документ - это счёт за потраченное электричество филиала Fandriana от компании электричества.

Из этого документа необходимо вводить в бухгалтерский отчёт интересующую нас информацию, т.е. сумму денег, списанную со счета филиала вместе с описанием и датой списания.

Иzm.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						10

	JIRO SY RANON'I FISAKANA "JIRAFI"				
BP 32 - Ilebona - FIANARANTSOA					
Tél. 033 22 10 200 - 201 - 202 - 203 - 204 Antananarivo					
BOA : 1 100 585 000					
FACTURE DE CONSOMMATION N° U/10449					
Mois de : DÉCEMBRE 2011					
Fait le :	30/09/2011	Nom :	RAHERY VOLOLONA Manitra		
CIN :	406551	Code :	5 Adresser		
Neufvilles index		1910	Consommation	Prix du kWh	Montant
Ancien index		1906		402	
↓ Montant consommation					5 106
↓ Abonnement					2 700
↓ Prime fixe					540
↓ Taxe communale					115
↓ FME pour la consommation supérieure à 20 Kwh					0
↓ TVA pour la consommation supérieure à 100 Kwh					0
TOTAL DE LA FACTURATION DU MOIS					6 861
Arrêtez la postule fiafia à la somme de :					
neuf mille six cent vingt six Ariary					
Votre situation :					
Antécédent :		0	 <div style="margin-top: 10px;">LE DIRECTEUR GÉNÉRAL <i>[Signature]</i></div>		
Facture du mois :		9 628			
Total à payer :		9 628			
Messages :					
Ny fandahana ny vato-pis dia hira (20) amin'ny asanefy ny rahaonan'i by Miasotsa Tompolika					
QUITTAANCE					
de la facture n°					
Mois de : DÉCEMBRE 2011					
Fait le : 10/02/2012					
 <div style="margin-top: 10px;">Ratsiraka, 10/02/2012 <i>[Signature]</i> RAVOSA Charline</div>					

Date	L'effacement de l'opération		Valeur	Date	Crédit
		Soit le 31/12/2013			
24.11	ADON DU JOURNAL AU 31/12/1975	31.12	2.840,46		
15.11	ABONNEMENT A LA PRESSE	31.12	1.000,00		
15.11	ABONNEMENT GY-HEMISTE	31.12	740,00		
16.11	ABONNEMENT A LA PRESSE	31.12	1.000,00		
16.11	DÉDUCTION MUSICA	31.12	2.000,00		
16.11	MALIN NOUVEAU 2013	31.12	1.404,00		
27.11	DECOUVERT-BANQUE/HOUSING	31.12	1.000,00		
	Total des mouvements		1.000,00		
	Soit au 29/12/2013		1.404,00		

Pour remettre votre compte 249124, 7,27 appeler VOCALIA au 023800 ou de votre poste directement avec les numéros abrégés

Рисунок 1.2- а) Квитанция от компании электричества о затратах филиала за декабрь 2011 г. б) Банковская выписка от банка BFV-SG Madagascar в ноябре 2013 г, содержащая транзакции по банковскому счету бухгалтера Raherivololona M.C.

Алгоритм распознавания букв и текста существует давно, но его применение в практической деятельности не настолько широко распространено, как хотелось бы. На рынке существует много приложений для распознавания текстов, но их сложно применять в реальной деятельности компаний, так как они только читают обычные тексты и переводят считанные данные в текстовой документ или записывают текст в буфер обмена для дальнейшего использования. Эти приложения не умеют распознавать структурированные документы или, точнее, не предназначены для структурированных документов (паспорт, выписки банковских транзакций и др.), не способны выделить и извлечь ключевую информацию, не обрабатывают считанные данных, а являются всего лишь доказательством того, что человечество уже открыло алгоритмы распознавания текстов. Распознавание структуры, выделение ключевой информации и обработка данных из документов на твёрдых носителях являются дополнительными задачами в том случае, если мы хотим, чтобы технологии распознавания текста эксплуатировались и использовались на практике.

						Лист
Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	11

1.2. Постановка задачи

Программа для оцифровки должна быть представлена в виде компьютерного приложения. Приложение должно обеспечивать распознавание табличных структур и форматов текстов, определённых пользователем, на входном изображении. Изображение должно находиться на локальном диске/носителе.

После обработки изображения и распознавания текстов, обрабатываются полученные структурированные данные, извлекается нужная информация и записывается в файл в выбранном пользователем формате.

При больших количествах входных изображений приложение должно обеспечивать автоматическую обработку всех или некоторых изображений, выбранных пользователям, или всех изображений, находящихся в выбранной пользователем папке.

1.3. Определение предметной области дипломного проекта

1.3.1. Исходные данные: изображение

В этом проекте входными данными являются **цифровые изображения**. Существует три основных способа цифрового представления изображений: растровая графика, векторная графика, фрактальная графика. Под цифровым изображением в данной работе понимается **растровая графика** - изображение, представляющее собой сетку пикселей или цветных точек. Это изображение можно получить при сканировании бумажного документа с использованием профессионального сканера. Независимо от способа получения растровое изображение должно иметь следующие характеристики):

Разрешение: каждая буква должна иметь разрешение не меньше 20x20px. В обычных документах формата А4 (210x297 мм) в распечатанном тексте с обычными шрифтами (Time New Roman, Arial, Courier New, размерность шрифта от 9 до 12) буквы в нижнем регистре занимают около 30x30px при сканировании документов профессиональным сканером и около

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						12

25x25px если документ был сфотографирован с использованием камеры с разрешением 8 мегапикселей.

Формат: исходный файл должен иметь один из форматов изображения, по умолчанию поддерживаемых на языке Java. В состав этих форматов входят GIF, PNG, JPEG, BMP и WBMP.

Цветовая модель пикселей: RGB (Red – Green – Blue) или ARGB (Alpha – Red - Green-Blue) с глубиной не менее 8-битов. Цветовая модель — математическая модель описания представления цветов в виде кортежей чисел (обычно из трёх, реже — четырёх значений), называемых цветовыми компонентами или цветовыми координатами ^[4]. RGB-модель описывает цвет тремя числами, соответствующими трем базовым цветам: красный, зелёный, синий. При смешении этих цветов можно получить X^3 разных цветов, где $X = 2^k$ и k - глубина пикселей. Таким образом при самой распространённой глубине пикселей (8 пикселей), пиксель в RGB-модели может принять до $(2^8)^3 = 256^3 = 16777216 > 16$ миллионов цветов.

1.3.2. Фильтрация

Обработка изображений может производиться в различных целях: изменение (искажение) изображения с целью достижения каких-либо эффектов (художественное улучшение); визуальное (заметное глазом) улучшение качества изображения (коррекция яркости и контраста, цветокоррекция и т.п.), объективное улучшение качества изображения (устранение искажений типа дисторсия, смазывание, расфокусировка и т.п.); проведение измерений на изображении (анализ интерферограмм, гарманограмм и т.п.); распознавание образов (распознавание символов, отпечатков пальцев, лиц, приборов наведения и т.п.) и т.п ^[5].

В данном проекте, для того чтобы распознать табличную структуру, необходимо пройти несколько этапов обработки изображений. Эти этапы нужны для устранения шумов и выделения/подчеркивания линий, образующих таблицы. Процесс устранения различных видов шумов на

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						13

изображениях называется **фильтрацией**. При осуществлении фильтрации яркостные характеристики каждой точки цифрового изображения заменяются другим значением яркости, которое признается в наименьшей степени искаженным помехой. Выделяют частотную и пространственную фильтрацию^[6]. В данной работе рассматривается только **пространственная фильтрация**.

Пространственные методы улучшения изображений применяются к растровым изображениям, представленным в виде двумерных матриц. Принцип пространственных алгоритмов заключается в применении специальных операторов к каждой точке исходного изображения. В качестве операторов выступают прямоугольные или квадратные матрицы, называемые **масками или ядрами**^[7]. Чаще всего маска представляет собой небольшой двумерный массив, а методы улучшения, базирующиеся на таком подходе, часто называют обработкой по маске или **фильтрацией по маске**.

При осуществлении линейной фильтрации отклик маски задается суммой произведений пикселей в области покрытия фильтра. Обычно произведение производится от верхней левой части изображения, слева направо, сверху вниз, при котором результат произведения заменяет значения, находящиеся на верхних левых пикселях замаскированной части изображения.

1.3.3. Обработка изображения детектором границ Кенни (Canny)

Края (границы) — это такие кривые на изображении, вдоль которых происходит резкое изменение яркости или других видов неоднородностей. Иными словами, край — это резкий переход/изменение яркости.

Причины возникновения краёв:

- изменение освещенности
- изменение цвета
- изменение глубины сцены (ориентации поверхности)

Получается, что края отражают важные особенности изображения, и поэтому целями преобразования изображения в набор кривых являются:

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						14

- выделение существенных характеристик изображения
- сокращение объема информации для последующего анализа

Самым популярным методом выделения границ является детектор границ Кенни. Хотя, работа Кенни была проведена на заре компьютерного зрения (1986), детектор границ Кенни до сих пор является одним из лучших детекторов [8].

Оператор Кэнни (детектор границ Кэнни, алгоритм Кэнни) в дисциплине компьютерного зрения — оператор обнаружения границ изображения.

Кэнни изучил математическую проблему получения фильтра, оптимального по критериям выделения, локализации и минимизации нескольких откликов одного края. Он показал, что искомый фильтр является суммой четырёх экспонент. Он также показал, что этот фильтр может быть хорошо приближен первой производной Гауссиана. Кэнни ввёл понятие подавления немаксимумов (англ. Non-Maximum Suppression, которое означает, что пикселями границ объявляются пиксели, в которых достигается локальный максимум градиента в направлении вектора градиентах [9].

Не смотря на то, что его работа была проведена на заре компьютерного зрения, детектор границ Кэнни до сих пор является одним из лучших детекторов. Кроме особенных частных случаев трудно найти детектор, который бы работал существенно лучше, чем детектор Кэнни.

1.3.4. Обработка изображений преобразованием Хафа

Преобразование Хафа (Hough Transform) — алгоритм, численный метод, применяемый для извлечения элементов из изображения. Используется в анализе изображений, цифровой обработке изображений и компьютерном зрении. Предназначен для поиска объектов, принадлежащих определённому классу фигур, с использованием процедуры голосования. Процедура голосования применяется к пространству параметров, из которого и получаются объекты определённого класса фигур по локальному максимуму

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						15

в так называемом накопительном пространстве (accumulator space), которое строится при вычислении трансформации Хафа [10].

Классический алгоритм преобразования Хафа связан с идентификацией прямых в изображении, но позже алгоритм был расширен возможностью идентификации позиции произвольной фигуры, чаще всего эллипсов и окружностей.

При автоматизированном анализе цифровых изображений очень часто возникает проблема идентификации простых фигур, таких как прямые, круги или эллипсы. Во многих случаях используется алгоритм поиска границ в качестве предобработки для получения точек, находящихся на кривой в изображении. Однако, либо из-за зашумлённости изображения, либо из-за несовершенства алгоритма обнаружения границ, могут появиться «потерянные» точки на кривой, также как и небольшие отклонения от идеальной формы прямой, круга или эллипса. По этим причинам часто довольно сложно приписать найденные границы соответствующим прямым, кругам и эллипсам в изображении. Назначение преобразования Хафа — разрешить проблему группировки граничных точек путём применения определённой процедуры голосования к набору параметризованных объектов изображения.

Прямая может быть задана уравнением $y = mx + b$ и может быть вычислена для любой пары точек на изображении (x, y). Главная идея в преобразовании Хафа — учесть характеристики прямой не как точек изображения, а в терминах её параметров, то есть m — коэффициента наклона и b — точки пересечения. Основываясь на этом факте, прямая $y = mx + b$ может быть представлена в виде точки с координатами (b, m) в пространстве параметров. Однако прямые, параллельные осям координат, имеют бесконечные значения для параметров m и b . Поэтому удобней представить прямую с помощью других параметров, известных как r и θ . θ - изменяется в пределах от 0 до 2π . r - ограничено размерами входного изображения.

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						16

При таком описании прямых не возникают бесконечные параметры.

1.3.5. Бинаризация изображения

Бинаризация изображения (*document image binarization*) - это процесс, который обычно применяется перед оптическим распознаванием символов. Суть этого процесса заключается в преобразования цветного изображения в чисто чёрно-белое изображение без оттенка серого. Т.е. после применения алгоритма бинаризации над исходным изображением, выходное изображение представляется только чёрными ($rgb(0,0,0)$) и белыми ($rgb(255, 255, 255)$) пикселями. Таким образом, изображение можно представить только бинарным массивом, что чрезмерно упрощает процесс обучения нейронной сети, а также распознавания букв в документах.

Хотя, алгоритм бинаризации изображений уже изучался в течении многих лет, нахождение общего идеального порога для любого искаженного изображения до сих пор остаётся нерешённой проблемой. Это объясняется тем, что моделирование переднего плана и фонового плана документа в изображении является затруднительной задачей из-за различных видов деградации документов, таких как неравномерное освещение, резкое изменение яркости изображения и пятно.

1.3.6. Алгоритм кластеризации K-Means

Кластерный анализ — многомерная статистическая процедура, выполняющая сбор данных, содержащих информацию о выборке объектов, и затем упорядочивающая объекты в сравнительно однородные группы. Задача кластеризации относится к статистической обработке, а также к широкому классу задач обучения без учителя [11].

K-Means (метод k-средних) — наиболее популярный метод кластеризации. Целью этого алгоритма является разделение некоторые наблюдений на k кластеров, при этом каждое наблюдение относится к тому кластеру, к центру (центроиду) которого оно ближе всего.

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						17

1.3.7. Градиентный спуск и его применение в нейронных сетях

Градиентный спуск — метод нахождения локального экстремума (минимума или максимума) функции с помощью движения вдоль градиента. Для минимизации функции в направлении градиента используются методы одномерной оптимизации, например, метод золотого сечения. Также можно искать не наилучшую точку в направлении градиента, а какую-либо лучше текущей.

Метод градиентного спуска с некоторой модификацией широко применяется для обучения перцептрона и в теории искусственных нейронных сетей известен как метод обратного распространения ошибки [12]. При обучении нейросети типа «персепtron» требуется изменять весовые коэффициенты сети так, чтобы минимизировать среднюю ошибку на выходе нейронной сети при подаче на вход последовательности обучающих входных данных. Формально, чтобы сделать всего один шаг по методу градиентного спуска (сделать всего одно изменение параметров сети), необходимо подать на вход сети последовательно абсолютно весь набор обучающих данных, для каждого объекта обучающих данных вычислить ошибку и рассчитать необходимую коррекцию коэффициентов сети (но не делать эту коррекцию), и уже после подачи всех данных рассчитать сумму в корректировке каждого коэффициента сети (сумма градиентов) и произвести коррекцию коэффициентов «на один шаг». Очевидно, что при большом наборе обучающих данных алгоритм будет работать крайне медленно, поэтому на практике часто производят корректировку коэффициентов сети после каждого элемента обучения, где значение градиента аппроксимируются градиентом функции стоимости, вычисленном только на одном элементе обучения. Такой метод называют стохастическим градиентным спуском или оперативным градиентным спуском. Стохастический градиентный спуск является одной из форм стохастического приближения. Теория стохастических приближений даёт условия сходимости метода стохастического градиентного спуска.

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						18

1.3.8. Нейронные сети и Метод обратного распространения ошибки

После выделения линии и распознавания структуры таблицы по выделенным линиям дальше обрабатывается каждая ячейка. Каждая ячейка таблицы изображения обрабатывается как отдельное изображение. Из каждой ячейки распознаются тексты. Для этого нужно выделить каждую букву, распознать и классифицировать её с помощью алгоритма распознавания букв. Одним из самых надежных и самых распространённых методов решения такой задачи является **метод обратного распространения ошибки (МОР)**. МОР - это один из алгоритмов обучения **искусственных нейронных сетей (ИНС)**. Идея искусственных нейронных сетей возникла как попытка описать процессы восприятия информации, происходящие в мозге человека. Как и мозг человека, ИНС состоит из множества соединенных друг с другом элементов – **перцепtronов**, которые имитируют нейроны головного мозга.

Перцептрон (*perceptron* от латинского слова *perceptio* — восприятие) — это математическая и компьютерная модель восприятия информации мозгом, предложенная Фрэнком Розенблаттом в 1957 году [13]. Перцептрон стал одной из первых моделей ИНС и состоит из трёх типов элементов: **сенсорные элементы** (S-элементы, перцепторы), **ассоциативные элементы** (A-элементы) и **реагирующие элементы** (R-элементы). Поступающие от сенсоров сигналы передаются ассоциативным элементам, а затем реагирующему элементам. Таким образом, перцептроны позволяют создать набор «ассоциаций» между входными стимулами и необходимой реакцией на выходе.

В принципе, МОР - это **итеративный градиентный алгоритм**, который используется с целью минимизации ошибки работы многослойного перцептрана и получения желаемого выхода. Основная идея этого метода состоит в распространении сигналов ошибки от выходов сети к её входам в направлении, обратном прямому распространению сигналов в обычном режиме работы.

Изм.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						19

1.3.9. Сохранение и обработка данных

После распознавания текстов и структуры таблицы необходимо обработать данные и вывести их в один из следующих форматов: XML-файл, CSV-файл, SQL-скрипт, или автоматически занести в базу данных.

CSV-файл - это один из самых распространённых форматов, используемых как альтернатива для сохранения табличных данных в обычных текстовых форматах, с расширением .csv. Каждая строка файла — это одна строка таблицы. Строки разделяются парой символов CR LF в DOS и системах Windows, а LF в UNIX-системах. Значения отдельных колонок разделяются разделительным символом — запятой (,) или точкой с запятой (;).

SQL-скрипт - это текстовой файл, содержащий последовательность SQL-запросов или SQL-команды для работы с базой данных. В данном проекте в качестве СУБД выбирается MySQL (версия 5.0 или более поздняя).

Иzm.	Лист	№ докум.	Подпись	Дата	ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ	Лист
						20

2. Проектирование программного обеспечения

2.1. Алгоритмы предобработки и отчистки изображения

Большинство методов анализа больших объемов данных требуют предварительной обработки входных данных с целью улучшения эффективности работы с ними. В данном проекте в качестве входных данных используется изображение. Рисунок, полученный в результате сканирования документа, обычно содержит более миллиона пикселей. Каждый пиксель занимает 4 байта. Следовательно, объем входящей информации составляет около 3 мегабайт. Большая часть приложений, занимающихся распознаванием текстов, пытается упростить задачу распознавания путем предобработки изображения. Методы и алгоритмы предобработки, которые используются в этом проекте, описаны ниже.

2.1.1. Уменьшение размерности входных данных

Прежде всего, необходимо попытаться уменьшить размер входных данных. Например, если приложение распознает текст, нет необходимости хранить цветной рисунок, так как буквы распознаются по своим формам.

Алгоритмы преобразования изображения в чёрно-белое

При преобразовании изображения в чёрно-белое размер входных данных значительно уменьшается. Для изображения без **альфа-канала**, таких как изображения формата *jpeg*, он уменьшается в 3 раза, а у изображения с байтом прозрачности – в 4 раза. Для преобразования изображения в чёрно-белое есть 3 типа алгоритмов, один из которых пользователь может самостоятельно выбрать в настройках программы. Каждый из этих вариантов использует все 3 цветовых компонента (красный-зелёный-синий) пикселей изображения, чтобы получить одно число от 0 до 255, означающее оттенок серого для выходных пикселей.

Иzm.	Лист	№ докум.	Подпись	Дат	ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ		
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	Лит.	Лист
Руковод.	Брусенцев А.И.					21	Листов
Консул.							
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.						
							БГТУ им. В. Г. Шухова, ПВ-51

Каждый пиксель обрабатывается отдельно. Предполагая, что компоненты пикселей больше одного, где r , g , b - это значения красного, зелёного и синего компонента соответственно, задаем формулы для получения оттенка серого b для этого пикселя для каждого из методов следующим образом:

1. $b = \frac{\max(r,g,b) + \min(r,g,b)}{2}$ – lightness grayscale
2. $b = \frac{r+g+b}{3}$ – average grayscale
3. $b = 0.21 * r + 0.72 * g + 0.07 * b$ – luminosity grayscale



Рисунок 2.1- Преобразование образца паспорта гражданина РФ в чёрно-белое с использованием трёх методов: a) lightness b) average c) luminosity

Выбор метода преобразования исходного изображения в чёрно-белое осуществляется пользователь, так как результат этого процесса очень важен для следующего шага (см. рисунок 2.1). Более предпочтительным является вариант, при котором оттенок текста сильно отличается от фонового. В таком случае самым эффективным методом является третий метод, поэтому по умолчанию выбирается именно он. Однако существуют случаи, при которых выбор другого метода преобразования цветного изображения в чёрно-белое является более целесообразным.

Алгоритм кластеризации K-Means

K-means clustering, или метод К-средних, - это самый популярный и простой метод кластеризации. Суть метода состоит в кластеризации объектов таким образом, чтобы расстояние между элементом и центром текущего

Изм.	Лист	№ докум.	Подпись	Дата

кластера было меньше, чем расстояние от этого элемента до центра любого другого кластера. Иными словами, действие алгоритма таково, что он стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров. Математически это можно описать следующим образом:

$$S_k = \sum_{i=1}^k \sum_{x_j \in S_i} (x_j - \mathbf{c}_i)^2 \rightarrow \min$$

Где:

k – число кластеров,

S_i – полученные кластеры,

\mathbf{c}_i – центр масс векторов $x_j \in S_i$

Алгоритм, позволяющий достичь данной цели, очень прост. Если необходимо разбить векторы на K кластеров, то:

- 1) Выбрать k произвольных разных векторов, как центр масс каждого кластера;
- 2) Пока есть кластер, у которого центр масс изменился (изменяется):
 - a) Запомнить центры масс K_i каждого i -го кластера;
 - b) Исключить все векторы из кластеров;
 - c) Для каждого вектора V :
 - i) Найти вектор K_i , евклидово расстояние до которого является минимальным;
 - ii) Включить V в i -ый кластер;
 - d) Найти новый центр масс K_i каждого i -го кластера;

Визуальный результат алгоритма K-means представлен на рисунке 2.2.

Недостатки метода K-means:

- Гарантируется достижение только одного из локальных минимумов, а не глобального минимума суммарного квадратичного отклонения $S_k = \sum_{i=1}^k \sum_{x_j \in S_i} (x_j - \mathbf{c}_i)^2$.
- Результат зависит от выбора исходных центров кластеров, их оптимальный выбор неизвестен.

Изм.	Лист	№ докум.	Подпись	Дата

- Число кластеров необходимо знать заранее.

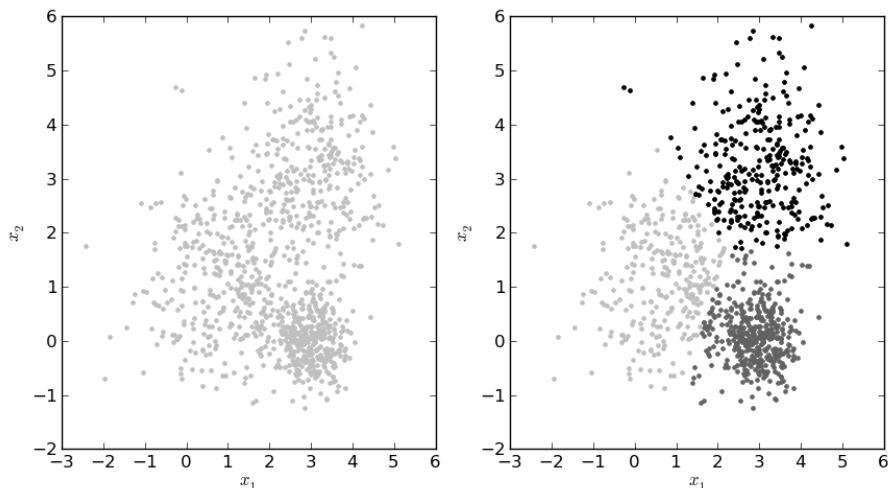


Рисунок 2.2 - Кластеризация набора пунктов по 3-м кластерам. До и после кластеризации.

Бинаризация изображения

Для превращения чёрно-белого изображения в бинарное существует множество алгоритмов. Каждый из них имеет свои достоинства и недостатки. Опытным путем для бинаризации был выбран наилучший с точки зрения качества распознавания текста алгоритм K-means. В качестве данных, подлежащих группировке, выступают пиксели чёрно-белого изображения. Здесь переменными являются векторы в 1-мерном пространстве, которые имеют значение $0 \leq b \leq 255$, т.е. последовательность чисел. Необходимо разбить пиксели на 2 группы: фоновые пиксели и тексты.

После нахождения кластеров методом K-means устанавливаем порог p для бинаризации.

$$p = \frac{\max(K_1) + \min(K_2)}{2} , \text{ где } \Pi_{K_1} < \Pi_{K_2}$$

Каждый пиксель чёрно-белого изображения затем заменяется на черный, если оттенок серого этого пикселя меньше p , иначе на белый, т.е.

$$\text{pix} \leftarrow \text{pix} > p ? 255 : 0$$

где:

pix - оттенок серого пикселя

Изм.	Лист	№ докум.	Подпись	Дата		ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	Лист 24

Так как алгоритм ищет локальный минимум, результат зависит от начальных значений центра масс (см. рисунок 2.3).

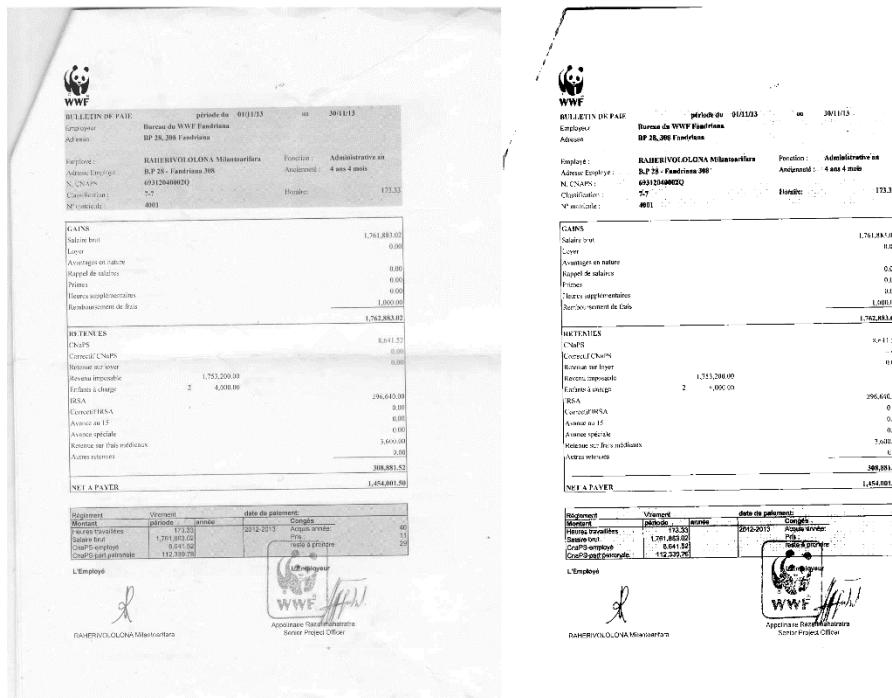


Рисунок 2.3- результат бинаризации отсканированной банковской выписки

2.1.2. Очистка изображения

Нас интересуют буквы и таблицы в изображении. Вся остальная информация является лишней и её необходимо удалить. Для этого необходимо найти все наборы пикселей, которые возможно представляют собой пиксели, отображающие буквы или таблицы. В русском и английском алфавите буквы не имеют разрывов, кроме буквы *i*, *j*, *й*, *ё*. Отсюда ясно, что смежные пиксели либо отображают букву или таблицу, либо шум. Следующий алгоритм используется для устранения шумов:

- 1) D_{in} – входной бинарное изображение
- 2) $\Omega \leftarrow \emptyset$ – совокупность всех набор пикселей отображающие буквы и таблицы
- 3) R – максимальное расстояние между двумя смежными пикселями
- 4) H_s - минимальная длина буквы в пикселях;

Изм.	Лист	№ докум.	Подпись	Дата

5) Для каждого пикселя P изображения:

- a) Если $P = \mathbf{0}$ (P - чёрный)
 - i) $FIFO \leftarrow \emptyset$
 - ii) $M \leftarrow \emptyset$ - набор смежных пикселей
 - iii) $x_{min} \leftarrow +\infty, x_{max} \leftarrow -\infty, y_{min} \leftarrow +\infty, y_{max} \leftarrow -\infty$ – края охватывающего прямоугольника набора смежных пикселей M
 - iv) Добавить P в $FIFO$
 - v) Пока $tmpM \neq \emptyset$
 - (1) Извлечь P из $FIFO$
 - (2) $P \leftarrow 1$ (чтобы к этому пикслю больше не обращались)
 - (3) $x_{min} \leftarrow \min(x_{min}, P_x)$
 - (4) $x_{max} \leftarrow \max(x_{max}, P_x)$
 - (5) $y_{min} \leftarrow \min(y_{min}, P_y)$
 - (6) $y_{max} \leftarrow \max(y_{max}, P_y)$
 - (7) Для всех пикселей Q, где $d(P, Q) = \|P - Q\| \leq R$ и $Q = 0$
 - (a) Добавить Q в $FIFO$
 - (8) Добавить P в M
 - vi) Если $|x_{min} - x_{max}| \geq H_s$ и $|y_{min} - y_{max}| \geq H_s$
 - (1) Добавить все пиксели из M в Ω
- 6) Создать новый документ D_{out} , размер которого равен размеру входного документа.
- 7) Для всех пикселей P из Ω :
 - a) Раскрасить пиксель с координатами (P_x, P_y) на документе D_{out} в чёрный цвет.
 - 8) Вывести D_{out} .

После применения этого алгоритма все излишние пиксели или наборы пикселей будут отфильтрованы. Изображение должно быть достаточно «чистым», чтобы перейти к следующему пункту.

Изм.	Лист	№ докум.	Подпись	Дата

2.1.3. Устранения наклона изображения

Безупречное сканирование документа человеком крайне маловероятно. Одной из самых очевидных ошибок является сканирование документа с наклоном. Люди очень редко обращают внимание на параллельность документа с краем сканера. Из-за этого полученное изображение имеет небольшой наклон. Для человеческого зрения, такие несовершенства не представляют проблем, а для компьютера это достаточно большое препятствие. Например, если буква в конце одной строки находится ниже/выше буквы в начале строки, установить, что они являются буквами одной и той же строки неэффективно и сложно, так как их у-координаты могут сильно отличаться и в результате мы получим плохо распознанные буквы в неупорядоченных строках.

Во избежание подобных проблем нужно исправить наклон документа.

Алгоритмы обработки изображения линейными операторами

Самым распространённым методом обработки изображений является метод линейной фильтрации. Применение свёртки разных ядер или операторов с изображением приведёт к разным интересным выходным результатом. Пиксели выходного изображения можно получить по следующей формуле [14]:

$$\text{Вых}(i, j) = \sum_{k=1}^m \sum_{l=1}^n \text{Bx}(i + k - 1, j + l - 1) * \text{Ядр}(k, l)$$

где:

i, j, m, n, M, N – числа, где $1 \leq i \leq M - m + 1, 1 \leq j \leq N - n + 1$,

$\text{Вых}(i, j), \text{Bx}(i, j)$ - пиксели с координатами (i, j) выходного и входного изображения соответственно;

Ядр – ядро фильтрации, т.е. матрица размерности $m \times n$;

Если описать псевдокодом

- 1) I – матрица пикселей входного изображения, размер $M \times N$
- 2) O – матрица выходного изображения, размер $M \times N$

Изм.	Лист	№ докум.	Подпись	Дата	ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	Лист
						27

- 3) K – линейный оператор, матрица $m \times n$
- 4) Для всех пикселей $I(i,j)$:
 - a) $T \leftarrow$ Подматрица размера $m \times n$ матрицы I ,
где первый элемент $T(1,1) = I(i,j)$;
 - b) $O(i,j) \leftarrow \sum_{1 \leq i' \leq m, 1 \leq j' \leq n} T(i',j') * K(i',j')$ – свертка;
- 5) Вывод О;

Обнаружение границ методом Кэнни

Суть этого метода состоит в отображении границ каждого объекта изображения. Это полезно для того, чтобы найти в документе, особенно с толстыми линиями, где и как линии и совокупности границ пикселей наклоняются. Для этого метод Кэнни проходит через несколько шагов, описанных ниже [15].

Сглаживание с фильтрацией по Гауссу – это метод линейного оператора с оператором Гаусса $K_{\text{гаусса}}$, т.е. матрицей размером $(2k+1) \times (2k+1)$, элементы которой зависят от среднеквадратичного отклонения σ следующим образом:

$$K(i,j) = \frac{e^{(-\frac{(i-k-1)^2 + (j-k-1)^2}{2\sigma^2})}}{2\pi\sigma^2},$$

Например, оператор Гаусса с размерностью 5×5 и $\sigma = 1,3$ имеет следующее значение:

$$K = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Роль этого фильтра состоит в увеличении размытости изображения, поможет избавиться от остальных пятен и неравномерного цвета пикселей.

Поиск градиентов. Границы отмечаются там, где градиент изображения приобретает максимальное значение. Он может иметь различное

Изм.	Лист	№ докум.	Подпись	Дата

направление, поэтому алгоритм Кэнни использует четыре фильтра для обнаружения горизонтальных и вертикальных ребер в размытом изображении.

$$G = \sqrt{G_x^2 + G_y^2},$$

$$\theta = \arctan2(G_x, G_y),$$

Угол, полученный из этой формулы, затем округляется так, чтобы он был пропорционален $\frac{\pi}{4}$.

Истончение ребер. Этот шаг нужно пройти для удаления нежелательных побочных точек по краям в изображении. Алгоритм состоит в проведении над каждым пикселям изображения двух последовательных шагов:

- Сравнение резкости ребер текущего пикселя с резкостью ребер пикселей по направлению градиентов;
- Если резкость ребер этого пикселя больше, чем всех остальных резкости ребер других пикселей в локальном регионе, то он сохраняется, иначе удаляется;

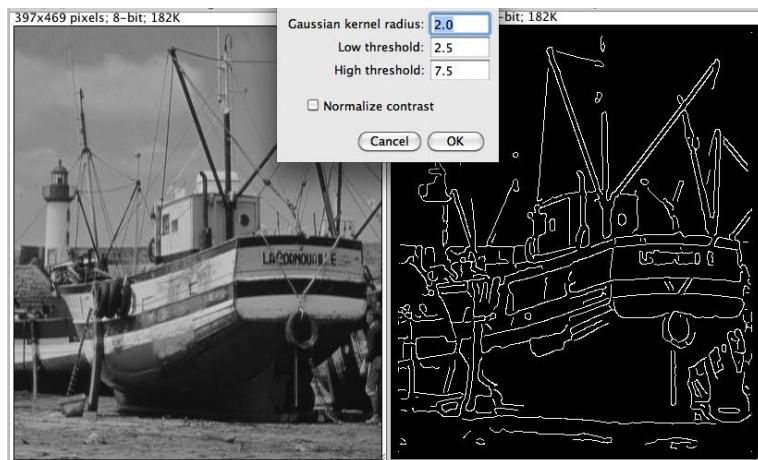


Рисунок 2.4 - Результат фильтрации Кэнни с оператором Гаусса размера 2, нижний порог 2.5 и верхний порог 7.5

Двойной порог. После применения третьего шага пиксели, представляющие собой края ребер, уже достаточно точные. Тем не менее, некоторые лишние пиксели ещё остались из-за шума и мелких изменений цвета. Для того, чтобы избавиться от них, можно ограничить значение

Изм.	Лист	№ докум.	Подпись	Дата

градиента: поставить минимальный и максимальный порог. Все значения градиента меньше минимального и больше максимального удаляются.

Пример результата применения всех этих шагов изображен на рисунке 2.4.

Преобразование Хафа

Преобразование Хафа, как уже было описано в предыдущей главе, помогает найти наборы пикселей, отображающие специальную фигуру (прямую или эллипс).

Математическое выражение для прямой можно представить в полярной форме (r, θ) (см. рисунок 2.5), где по аналогии с картезианским выражением имеем:

$$y = -\frac{\cos(\theta)}{\sin(\theta)} x + \frac{r}{\sin(\theta)},$$

то есть, $r = x \cos(\theta) + y \sin(\theta)$, (11).

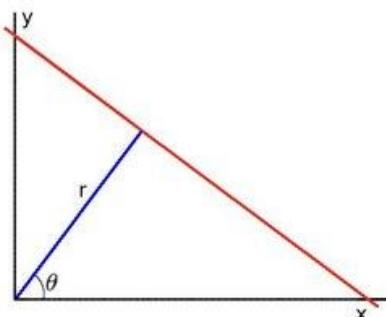


Рисунок 2.5 - Отображение линия с помощью полярного координата (r, θ)

Если задана точка (x_0, y_0) , то мы можем найти все пары (r_θ, θ) удовлетворяющие условию $r_\theta = x_0 \cos(\theta) + y_0 \sin(\theta)$. Это значит, что каждая пара (r_θ, θ) , которая является решением этого уравнения, представляет собой прямые. Решение этого уравнения есть совокупность всех прямых, проходящих через точку (x_0, y_0) [16].

Например, если задана точка $P(x, y), x = 8, y = 6$, то пары решения можно представить в виде синусоидальной кривой линии в пространстве Хафа как на рисунке 2.6.

Изм.	Лист	№ докум.	Подпись	Дата

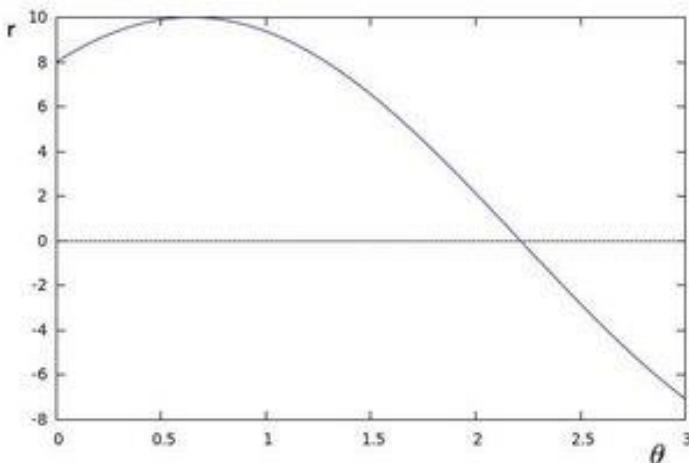


Рисунок 2.6 - Пары решения (r, θ) , при $x = 8, y = 6$, в пространстве Хафа

Из этого следует, если кривые, представляющие собой решения двух разных пунктов (x_0, y_0) и (x_1, y_1) пересекаются, то точка пересечения (r_0, θ_0) означает, что прямая $y = -\frac{\cos(\theta_0)}{\sin(\theta_0)} x + \frac{r_0}{\sin(\theta_0)}$ проходит через оба пункта.

На следующем рисунке представляются кривые, которые обозначают пары решения (r, θ) для пунктов $A(8,6), B(9,4), C(12,3)$ (см. рисунок 2.7).

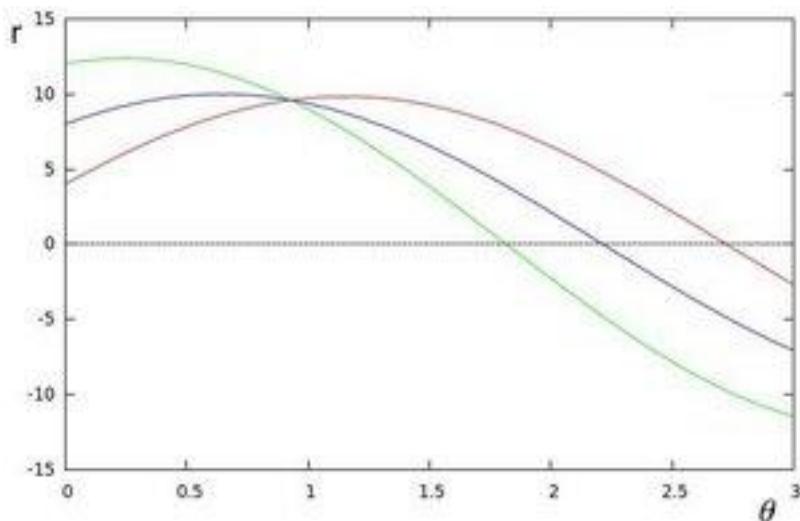


Рисунок 2.7- Кривые для пунктов A, B и C в пространстве Хафа

Естественно, эти 3 точки лежат на одной и той же прямой, поэтому их соответствующие кривые в пространстве Хафа пересекаются в одном и том же пункте.

После применения метода Кэнни, все ребра отображаются светлыми пикселями, а остальные тёмными пикселями. Поэтому, алгоритм

Изм.	Лист	№ докум.	Подпись	Дата

преобразование Хафа рисует соответствующие кривые в пространстве Хафа каждого светлого пункта и найдёт все пункты в этом пространстве, где пересекаются большое количество кривых. Определяется порог p количество пересекающих кривых. Если пересекаются больше чем p кривых в одном пункте (r_0, θ_0) , то считается, что есть линии с уровнем $y = -\frac{\cos(\theta_0)}{\sin(\theta_0)}x + \frac{r_0}{\sin(\theta_0)}$ в исходном изображении. Координаты линии сохраняются, чтобы определить наклон документа.

Определение и устранение наклона

Алгоритм Кэнни и преобразования Хафа поможет найти не только линию, но также и наклон строки. Экспериментирование с различными наборами параметров позволяет сконфигурировать алгоритмы таким образом, чтобы в результате они выводили линии, проходящие через строки как на рисунке 2.8.

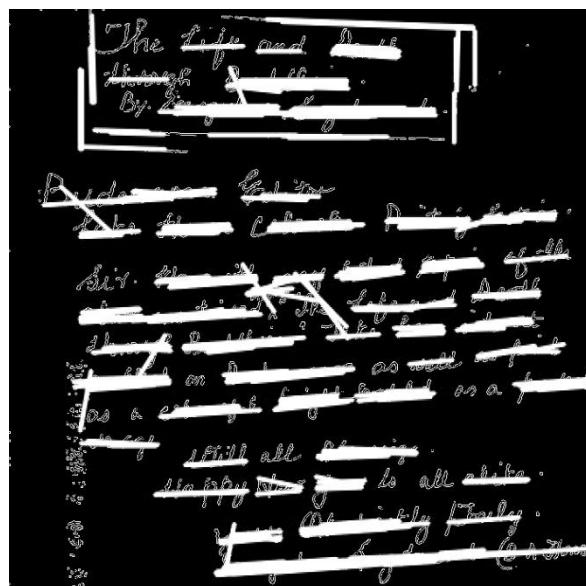


Рисунок 2.8 - Результат анализа рукописного письма после применения метода Кэнни и преобразования Хафа

Таким образом, зная линии, проходящие через строки, можно найти среднее значение отклонения всех линий и выводить приблизительный наклон документа.

Изм.	Лист	№ докум.	Подпись	Дата

2.1.4. Обнаружение таблицы

Таблица – это самая большая структура, которая может находиться в документе (если известно, что рисунков в документе нет).

В предыдущем этапе при отчистке документа были найдены наборы всех смежных пикселей. Теперь необходимо лишь пересмотреть отдельно каждый полученный набор и вывести в новый чистый документ тот, охватывающий квадрат которого имеет максимальную площадь. Снова применить Кэнни и Хаф в этом документе и сохранить информацию об обнаруженных отрезках линии.

Удаление излишних линий

После преобразования Хафа обнаруженные линии далеко не совершенны, они могут быть разбитыми и направленными в разные стороны, поэтому нужно применить дополнительные шаги. Следующий алгоритм определяет, можно ли слить два сегмента, чтобы получить один и выводит результат слияния. В противном случае выводится *null*.

- 1) Пусть L_1, L_2 два отрезка;
- 2) Пусть $x_{min1}, x_{max1}, y_{min1}, y_{max1}$ – минимальный и максимальный из x и y координат края сегмента L_1 ;
- 3) Пусть $x_{min2}, x_{max2}, y_{min2}, y_{max2}$ – минимальный и максимальный из x и y координат края сегмента L_2 ;
- 4) **Если угол отклонения между двумя линиями слишком большой, то алгоритм завершается.** Выводить пустой результат.
- 5) **Иначе:**
 - a) Пусть u – максимальное допустимое расстояние между краями двух смежных линий
 - b) Пусть v – максимальное допустимое расстояние между параллельными линиями
 - c) **Если L горизонтальные (изменение в x-координат больше чем в y-координат):**

Изм.	Лист	№ докум.	Подпись	Дата	ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	Лист
						33

i) Если расстояние между линиями больше допустимого, то есть,

$\left| \frac{y_{min1} + y_{max1}}{2} - \frac{y_{min2} + y_{max2}}{2} \right| > v$, то алгоритм завершается. Вывод пустой.

ii) Иначе:

(1) Если расстояние между краями больше допустимого, то есть

$x_{max1} + u < x_{min2}$ или $x_{max2} + u < x_{min1}$, то алгоритм завершается. Вывод пустой

(2) Иначе, выводить отрезок P_1P_2 , P1 и P2 края нового отрезка, где:

$$P_1 = (\min(x_{min1}, x_{min2}), \frac{y_{min1} + y_{max1} + y_{min2} + y_{max2}}{4})$$

$$P_2 = (\max(x_{max1}, x_{max2}), \frac{y_{min1} + y_{max1} + y_{min2} + y_{max2}}{4})$$

d) Если L вертикальные (изменение в x-координат меньше чем в y-координат):

i) Если расстояние между линиями больше допустимого, то есть,

$\left| \frac{x_{min1} + x_{max1}}{2} - \frac{x_{min2} + x_{max2}}{2} \right| > v$, то алгоритм завершается. Вывод пустой.

ii) Иначе:

(1) Если расстояние между краями больше допустимого, то есть

$y_{max1} + u < y_{min2}$ или $y_{max2} + u < y_{min1}$, то алгоритм завершается. Вывод пустой

(2) Иначе, выводить отрезок P_1P_2 , P1 и P2 края нового отрезка, где:

$$P_1 = (\frac{x_{min1} + x_{max1} + x_{min2} + x_{max2}}{4}, \min(y_{min1}, y_{min2}))$$

$$P_2 = (\frac{x_{min1} + x_{max1} + x_{min2} + x_{max2}}{4}, \max(y_{max1}, y_{max2}))$$

Теперь, задача слияния всех близких и смежных – тривиальная:

- 1) $\Gamma \leftarrow$ множество отрезков получены от преобразования Хафа
- 2) $\Delta \leftarrow \emptyset$, множество результатов слияний близких и смежных отрезков
- 3) **Пока** $\Gamma \neq \emptyset$:

Изм.	Лист	№ докум.	Подпись	Дата

- a) Извлекать один отрезок P из Γ
 - b) Если найдётся отрезок Q , с которым можно сливать P , то
 - i) $T \leftarrow$ слияние P и Q
 - ii) Удалить P и Q из множества Γ
 - iii) Добавить T во множество Γ
 - c) Иначе, добавить P в Δ
- 4) Выводить Δ

После применения этого алгоритма, количество отрезков на рисунке значительно уменьшится. При экспериментировании очень часто на выходе преобразования Хафа обнаруживается до 90 элементов отрезков, а после слияния всех близких и смежных отрезков остаются около 20. В результате получим чёткие вертикальные и горизонтальные отрезки, составляющие таблицу.

Обнаружение всех возможных клеток

Для обнаружения клеток надо сначала найти все пересечения отрезков. Это поможет найти уголки клеток. Результатом пересечений является множество пунктов. Эти пункты являются уголками клеток, и каждый пункт является верхним левым углом одной клетки. Процедура отображена на рисунке 2.9.

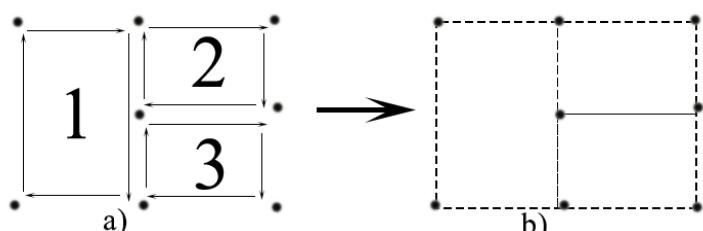


Рисунок 2.9- а) множество девяти точек, представляющих собой пересечения оставшихся отрезков. Если взять одну точку и найти три остальных точки, составляющих клетки по часовой стрелке, то в итоге получается клетка. б) Найденные клетки представляют собой таблицу.

2.2. Распознавание текстов

Единственным эффективным способом распознавания текстов является алгоритм нейронной сети. Этапы распознавания очевидные, хотя непростые.

Изм.	Лист	№ докум.	Подпись	Дата	ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	Лист 35

Сначала надо распознать все буквы в изображении, затем объединить их, чтобы получить слова и строки. Все эти процессы описаны внутри библиотеки Tesseract, о которой детально рассказывается в следующей главе. Основным алгоритмом в этой библиотеке является алгоритм распознавания текстов при помощи нейронных сетей.

2.2.1. Алгоритм метода обратного распространения ошибки

Метод обратного распространения (далее - МОР) фактически представляет собой обобщение метода наименьших квадратов применительно к многослойным нейронным сетям. Он с помощью простого градиентного спуска минимизирует среднеквадратичную ошибку между действительным и требуемым выходом нейронной сети [17].

Использование градиентного спуска предполагает наличие определённых требований к активационным функциям нейронов. Функция активации должна быть непрерывной, дифференцируемой и монотонно возрастающей. Одной из наиболее распространённых функций является бинарная сигмоидальная функция активации:

$$g(x) = \frac{1}{1 + e^{-x}},$$

При стремлении к $\pm\infty$, функция $g(x)$ стремится к 0 или 1. И её производная равна

$$g'(x) = g(x)(1 - g(x)),$$

В качестве примера для объяснения принципа действия алгоритма МОР будем использовать нейронную сеть, изображенную на рисунке 2.10.

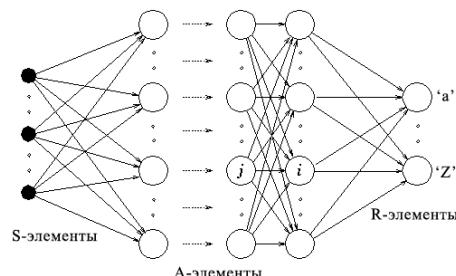


Рисунок 2.10 - Многослойная нейронная сеть

Изм.	Лист	№ докум.	Подпись	Дата

1) Все связи нейронной сети инициируются множеством небольших случайных значений.

2) Для каждой обучающей пары (S, T) (входное изображение, соответствующее значение)

a) Делаем следующее:

i) Этап прямой передачи:

(1) Пиксели входного бинарного изображения S разбиваются на S-элементы, где белые пиксели имеют значение 0, а чёрные – 1, и передаются A-элементам.

(2) Для каждого слоя L скрытых элементов:

(а) Для каждого A_j -элемента слоя L:

(i) $U_{\text{вх}} \leftarrow w_{0j} + \sum_{i=1}^n v_i * w_{ij}$ - входной сигнал элемента A_j где w_{ij} - весь связей между i-ого нейрона предыдущего слоя и j-ого нейрона текущего слоя, v_i - выходной сигнал i-ого элемента предыдущего слоя, n – количество A-элементов предыдущего слоя,

(ii) $U_{\text{вых}} \leftarrow g(U_{\text{вх}})$ – выходной сигнал элемента A_j .

ii) Этап обратного распространения ошибки:

(1) $E_i \leftarrow R_i - T_i$, разница между входным сигналом i-ого элемента последнего слоя и требуемым значением i-ого элемента.

(2) $\delta_i \leftarrow E_i * g'(R_i)$, ошибка i-ого элемента последнего слоя.

(3) Для каждого слоя L скрытых элементов начиная с последнего:

(а) Для каждого A_i -элемента слоя L:

(b) $\delta_{\text{вх},i} \leftarrow \sum_j \delta_j * w_{ij}$ – входное значение ошибки элемента A_j где w_{ij} - весь связей между A_i -элемента и j-ого нейрона следующего слоя δ_i - ошибка j-ого элемента следующего слоя.

(c) $\delta_i \leftarrow \delta_{\text{вх},i} * g'(U_{\text{вх}})$ – ошибка A_i -элемента.

iii) Этап коррекции весов и смещения:

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

(1) Для всех связей w_{ij} между i-ым элементом одного слоя и j-ым элементом следующего слоя:

- (a) $w_{ij} \leftarrow w_{ij} + \alpha U_{\text{вых},i} \delta_j$ – где: $0 \leq \alpha \leq 1$ – скорость обучения.
- b) Пока $\sum E_i^2 > \epsilon$ или цикл повторился больше M – раз
- c) Если $\sum E_i^2 > \epsilon$ то обучение успешно прошло, иначе невозможно изучать тренировочные данные.

При успешном завершении алгоритма с буквами одного алфавита нужно сохранить значения весов, так как они являются свойством этой нейронной сети. При необходимости использовать обученную нейронную сеть достаточно просто загрузить веса связей между элементами.

2.2.2. Распознавание букв

Чтобы распознавание было более эффективным, предпочтительно тренировать отдельные сети для каждого алфавита. Чем больше разнообразие тренировочных данных (шрифты, размеры, и т.д.), тем эффективнее будет распознавание. Крайне желательно, чтобы одинаковые буквы в тренировочных данных не слишком сильно отличались по фигуре, иначе тренировка может не состояться (ошибка в последнем слое не сойдется к значению меньше ϵ).

При распознавании букв нужно подключить соответствующую нейронную сеть. Алгоритм распознавание букв, использующий уже обученную сеть, имеет сложность по времени $O(1)$. Для распознавания букв в изображения используется скользящее окно (см. рисунок 2.11).



Рисунок 2.11 - Визуальное отображение работы алгоритм

Принцип работы скользящего окна простой:

Изм.	Лист	№ докум.	Подпись	Дата

1) Пусть h_{min}, h_{max} минимальный и максимальный размер окна в пикселях.

Только буквы, которые имеют такие размеры, будут участвовать в распознавании.

2) Для $h \leftarrow h_{min}, h_{max}$:

а) Двигать окно по документу с некоторым шагом и для каждой распознанной фигуры, сохранить позицию и соответствующий символ (буква/цифра/знак) данной фигуры.

Упорядочить распознанные буквы/цифры/знаки по их позиции в документе (слева направо, сверху вниз).

3) Вычислить среднее значение размера букв.

4) Если расстояние между двумя последовательными буквами больше среднего значения, то скорее всего это пробел. По найденным пробелам определить слова.

5) Вывести текст.

2.3. Исключение ключевых данных

Из распознанного текста вместе с таблицами необходимо найти ту информацию, которая интересует пользователя. Известно, что общая эффективность распознавания текстов большинства приложений в настоящее время ещё очень низка, даже с учетом трудоёмкой предобработки изображения, повышающей эффективность нейронной сети. Поэтому при работе с полученным при распознании текстом надо быть осторожным. Так как в качестве входных данных для этой программы выступает не только отсканированный документ, но также образец или формат документа, то нам надо найти и распознать этот образец в прочитанном тексте. Для этого используется метод нечёткого поиска строки.

2.3.1. Расстояние Левенштейна и нечёткий поиск строки

Нечёткий поиск строки находит подстроку или строку из множества строк, которая имеет максимальную схожесть с заданной строкой. Этую

Изм.	Лист	№ докум.	Подпись	Дата	ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	Лист
						39

схожесть можно оценивать разными способами. Самый простой способ - расстояние Левенштейна.

Расстояние Левенштейна определяет минимальное количество изменений, которые нужно применять в первой строке, чтобы получить вторую строку. Под изменением понимается удаление, добавление или изменение одной буквы [18].

Алгоритм нахождения расстояния Левенштейна является динамическим алгоритмом. Значит, в худшем случае он имеет сложность $O(n^2)$. В общем случае он работает достаточно быстро. Математически этот алгоритм описывается таким образом:

$$D^{Lev}_{a,b}(i,j) = \begin{cases} \max(i,j), & \text{если } \min(i,j) = 0 \\ \min \begin{cases} D^{Lev}_{a,b}(i-1,j) + 1 \\ D^{Lev}_{a,b}(i,j-1) + 1 \\ D^{Lev}_{a,b}(i-1,j-1) + 1_{a_i \neq b_j} \end{cases}, & \end{cases}$$

где:

$D^{Lev}_{a,b}(|a|, |b|)$ – расстояние Левенштейна между строкой а и б

$1_{a_i \neq b_j}$ – функция индикатора, равна 0 если $a_i \neq b_j$

Расстояние Левенштейна поможет особенно в проверке паттерна (шаблона) текстового документа и нахождении нужных пользователем переменных.

Чтобы найти подстроку с минимальным расстоянием Левенштейна к заданной строке применяется следующий алгоритм:

- 1) $\alpha \leftarrow$ заданная строка;
- 2) $S \leftarrow$ строка;
- 3) $d_{min} \leftarrow \infty$;
- 4) s_{min} , подстрока строки S с минимальным расстоянием к α ;
- 5) Для всех $0 \leq i < j \leq |S|$:
 - a) Если $D^{Lev}_{\alpha, S_{ij}} < d_{min}$, то:
 - i) $d_{min} \leftarrow D^{Lev}_{\alpha, S_{ij}}$;

Изм.	Лист	№ докум.	Подпись	Дата

- ii) $s_{min} \leftarrow S_{ij}$;
- 6) Выводить s_{min} ;

2.3.2. Обнаружение паттерна документа и исключение ключевых слов

Обнаружение табличного паттерна документа заключается только в нахождении всех клеток и запоминании позиции этих клеток (т.е. позиции строки и столбца).

Нечёткий поиск строки используется при обнаружении паттерна в текстовом документе. Паттерн составляется из строк и переменных (ключевой информации). Задача заключается в нахождении значения этих переменных в документе. Алгоритм решения этой задачи описан ниже:

- 1) $S \leftarrow$ считываемый текст из документа;
- 2) $F \leftarrow (\alpha_1, \gamma_1, \alpha_2, \gamma_2, \alpha_3, \gamma_3, \dots, \alpha_n, \gamma_n)$ – последовательность постоянных строк $(\alpha_1, \alpha_2, \dots)$ и переменных $(\gamma_1, \gamma_2, \dots)$;
- 3) $\Omega_{<\gamma,s>}$, множество пары $<\text{переменные}, \text{значение}>$;
- 4) Для всех i от 1 до $n-1$:
 - a) $u \leftarrow S_{p_1 q_1}$ подстрока строки S , с минимальным расстоянием к α_i ;
 - b) $v \leftarrow S_{p_2 q_2}$ подстрока строки S , с минимальным расстоянием к α_{i+1} ;
 - c) Увеличивать q_1 , пока S_{q_1} не пустой символ;
 - d) Уменьшать p_2 , пока S_{p_2} не пустой символ;
 - e) $s_i \leftarrow S_{q_1 p_2}$;
 - f) Отчистить строку s_i (удалить лишние пробелы);
 - g) Добавить пару (γ_i, s_i) в $\Omega_{<\gamma,s>}$;

Вывести $\Omega_{<\gamma,s>}$.

Изм.	Лист	№ докум.	Подпись	Дата

3. Программная реализация

3.1. Описание используемых инструментов и технологий

3.1.1. Библиотека OpenCV

OpenCV – (англ. Open Source Computer Vision Library) — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. OpenCV была реализована на C/C++, также разрабатывается для Python, Java, Ruby, Matlab, Lua и других языков, и содержит алгоритмы для интерпретации изображений, калибровки камеры по эталону, устранение оптических искажений, определение сходства, анализ перемещения объекта, определение формы объекта и слежение за объектом, 3D-реконструкция, сегментация объекта, распознавание жестов и т.д. Эта библиотека очень популярна за счёт своей открытости и возможности бесплатно использовать как в учебных, так и коммерческих целях [19].

У библиотеки OpenCV есть несколько модулей. В этом проекте в основном используется только один модуль: CV. Этот модуль заключает в себя:

- базовые операции над изображениями (фильтрация, геометрические преобразования, преобразование цветовых пространств и т. д.)
- анализ изображений (выбор отличительных признаков, морфология, поиск контуров, гистограммы)
- анализ движения, слежение за объектами
- обнаружение объектов, в частности лиц
- калибровка камер, элементы восстановления пространственной структуры

В этом проекте используются всего 2 метода этой библиотеки: метод Canny и метод HoughTransformP, так как все остальные нужные функции для работы с изображениями уже есть по умолчанию в Java Standard Edition 7.

Изм.	Лист	№ докум.	Подпись	Дат	ПРОГРАММНАЯ РЕАЛИЗАЦИЯ		
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	Лит.	Лист
Руковод.	Брусенцев А.И.				42	116	Листов
Консул.							
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.				БГТУ им. В. Г. Шухова, ПВ-51		

Метод Canny

Алгоритм обнаружение границ был описан в предыдущей главе. Метод Canny библиотеки OpenCV имплементирует все шаги алгоритма с некоторым улучшением. Вкратце, шаги, применяемые детектора:

- Убрать шум и лишние детали из изображения
- Рассчитать градиент изображения
- Сделать края тонкими (edge thinning)
- Связать края в контуре (edge linking)

Детектор использует фильтр на основе первой производной от гауссианы. Так как он восприимчив к шумам, лучше не применять данный метод на необработанные изображения. Сначала, исходные изображения нужно свернуть с гауссовым фильтром.

Границы на изображении могут находиться в различных направлениях, поэтому алгоритм Кенни использует четыре фильтра для выявления горизонтальных, вертикальных и диагональных границ. Воспользовавшись оператором обнаружения границ (например, оператором Sobel) получается значение для первой производной в горизонтальном направлении (G_y) и вертикальном направлении (G_x).

Из этого градиента можно получить угол направления границы:

$$Q = \arctan\left(\frac{G_x}{G_y}\right)$$

Угол направления границы округляется до одной из четырех углов, представляющих вертикаль, горизонталь и две диагонали (например, 0, 45, 90 и 135 градусов).

Затем идет проверка того, достигает ли величина градиента локального максимума в соответствующем направлении.

В OpenCV, детектор границ Кенни реализуется функцией `cvCanny()`, которая обрабатывает только одноканальные (чёрно-белое, или один из цвета каналов RGB) изображения.

Изм.	Лист	№ докум.	Подпись	Дата

Описание функции [20]:

```
CVAPI(void) cvCanny(const CvArr* image, CvArr* edges, double  
threshold1, double threshold2, int aperture_size CV_DEFAULT(3));  
  
- image — одноканальное изображение для обработки (градации  
серого)  
- edges — одноканальное изображение для хранения границ,  
найденных функцией  
- threshold1 — порог минимума  
- threshold2 — порог максимума  
- aperture_size — размер для оператора Sobel
```

3.1.1.1. Метод HoughTransformP

cvCanny() помогает замечательно выделять границы предметов и окружающей обстановки, поэтому он часто применяется до применения метода преобразования Хафа.

В двух словах, преобразование Хафа основывается на представлении искомого объекта в виде параметрического уравнения. Параметры этого уравнения представляют фазовое пространство (т.н. аккумуляторный массив/пространство, пространство Хафа).

Затем, берётся двоичное изображение (например, результат работы детектора границ Кэнни). Перебираются все точки границ и делается предположение, что точка принадлежит линии искомого объекта — т.о. для каждой точки изображения рассчитывается нужное уравнение и получаются необходимые параметры, которые сохраняются в пространстве Хафа.

Финальным шагом является обход пространства Хафа и выбор максимальных значений, за которые «проголосовало» больше всего пикселей картинки, что и даёт параметры для уравнений искомого объекта.

В OpenCV есть функция, реализующая преобразование Хафа для поиска линий. Описание этой функции следующее:

Изм.	Лист	№ докум.	Подпись	Дата

```
CVAPI (CvSeq*) cvHoughLines2 (CvArr* image, void* line_storage,  
int method, double rho, double theta, int threshold, double  
param1 CV_DEFAULT(0), double param2 CV_DEFAULT(0));
```

- *image* — 8-битное двоичное изображение (в случае вероятностного метода изображение будет модифицироваться)
- *line_storage* — хранилище памяти для сохранения найденных линий (можно использовать матрицу $1 \times \text{Число_линий} \text{ CvMat}$)
- *method* — метод: который принимает одно из следующих значений:
 - *CV_HOUGH_STANDARD* — классический вариант трансформации Хафа. Каждая линия представляется двумя числами типа *float* (*rho*, *theta*), где *rho* — дистанция между точкой (0,0) и линией, а *theta* — угол между осью x и нормалью к линии (т.е. матрица должна иметь тип of *CV_32FC2*)
 - *CV_HOUGH_PROBABILISTIC* — вероятностный метод трансформации Хафа (более эффективен с случае изображений с несколькими длинными линейными сегментами). Возвращает сегменты линии, которые представляются точками начала и конца (т.е. матрица должна иметь тип of *CV_32SC4*). Вероятностное преобразование Хафа (Probabilistic Hough Transform) заключается в том, что для нахождения объекта достаточно провести преобразование Хафа только для части (*a*) точек исходного изображения, $0\% \leq a \leq 100\%$. То есть сначала провести выделение «контрольных» точек с изображения, и для него провести преобразование Хафа.
 - *CV_HOUGH_MULTI_SCALE* — масштабный вариант классической трансформации Хафа. Линии представлены так же, как и в *CV_HOUGH_STANDARD*
- *rho* — разрешение по дистанции
- *theta* — разрешение по углу (в радианах)

Изм.	Лист	№ докум.	Подпись	Дата

- *threshold* — пороговый параметр. Линия возвращается, если акумулирующий параметр больше порогового значения.
- *param1* — первый параметр, в зависимости от метода трансформации:
 - *CV_HOUGH_STANDARD* — 0 — не используется
 - *CV_HOUGH_PROBABILISTIC* — минимальная длина линии
 - *CV_HOUGH_MULTI_SCALE* — делитель разрешения по дистанции
 $(\frac{\rho}{param1})$
- *param2* — второй параметр, в зависимости от метода трансформации:
 - *CV_HOUGH_STANDARD* — 0 — не используется
 - *CV_HOUGH_PROBABILISTIC* — максимальный промежуток между сегментами линии, лежащими на одной линии, чтобы считать их одним сегментом (объединить их вместе)
 - *CV_HOUGH_MULTI_SCALE* — делитель разрешения по углу
 $(\frac{\theta}{param1})$

Если *line_storage* — указатель на хранилище памяти, то функция вернёт указатель на первый элемент последовательности, содержащей найденные линии.

3.1.2. Библиотека Tesseract

Tesseract — свободная компьютерная программа для распознавания текстов, разрабатывавшаяся Hewlett-Packard с середины 1980-х по середину 1990-х, а затем 10 лет «пролежавшая на полке». В августе 2006 г. Google купил её и открыл исходные тексты под лицензией Apache 2.0 для продолжения разработки. В настоящий момент программа уже работает с UTF-8, поддержка языков (включая русский с версии 3.0) осуществляется с помощью дополнительных модулей [21]. Она считается одной из самых точных программ для распознавания текстов с открытым исходным кодом в настоящее время.

Изм.	Лист	№ докум.	Подпись	Дата

Так как этот проект написан на языке Java, и потому, что tesseract было написано на языке C++, необходимо использовать обёртку библиотеки написана на Java. Обёртка называется Tess4J.

Перед тем чтобы использовать функции этой библиотеки, надо создать объект *Tesseract*:

```
Tesseract tesseract = new Tesseract();
```

Все функции и параметризации дальше делается через этот объект. Список функций, которые были использованы [22]:

- *Tesseract.setLanguage(String lang)*: Указывает язык текста на документе или в изображении. Параметр *lang* – 3х-буквая строка, обозначающая название языка. “*eng*” – английский, “*rus*” – русский, “*fra*” – французский. По умолчанию, язык – английский.
- *Tesseract.setPageSegMode(int mode)*: Указывает метод работы
- *Tesseract.setTessVariable(String variable, String value)*: Установить значение внутреннего параметра *variable*.
- *Tesseract.doOCR(BufferedImage image)*: Распознавать текст из рисунка *image* и возвращает просчитанный текст.

3.2. Структура базы данных

Так как программный продукт предназначен для использования в разных документах, для каждого вида документа необходимо создать отдельную модель. Чтобы эти модели можно было использовать несколько раз, необходимо их сохранить. Итак, информация о модели документов, конфигурации и характеристики проектов сохраняются в базе данных.

В качестве СУБД было выбрано MySQL - свободная реляционная система управления базами данных. Она одна из наиболее популярных и эффективных систем управления базами данных, которая очень часто используется при построении не только современных веб-сайтов, а также для построения таких малых и средних приложений, как программный продукт этого проекта. СУБД MySQL поддерживает язык запросов SQL. Это позволяет

Изм.	Лист	№ докум.	Подпись	Дата

совершать такие операции, как запись данных в базу, редактирование данных, извлечение или удаление данных из базы данных.

3.2.1. Диаграмма сущности-связи

В состав этой базы данных входят следующие 6 таблиц, диаграмма сущности-связи которых представлена на рисунке 3.1:

- *OCRConfig* – таблица, в которой сохраняются параметры конфигурации распознавания текстов. Каждый столбец означает параметр для используемых функций в программе, кроме столбца *name*:
 - *name* – название конфигурации
 - *language* – язык документа
 - *grayscale* – метод для приведения изображения в чёрно-белое
 - *binarisation* – если true, то использовать метод K-Means для бинаризации, иначе используется $p = 128$ для порога бинаризации
 - *radius* – максимальное расстояние между чёрными пикселями, чтобы можно было их считать смежными.
 - *threshold1* – нижний порог детектора Кэнни
 - *threshold2* – верхний порог детектора Кэнни
 - *distanceAccumulation* - разрешение по дистанции преобразования Хафа
 - *angleAccumulation* - разрешение по углу преобразования Хафа
 - *thresholdAccumulation* - пороговый параметр преобразования Хафа
 - *minLength* - минимальная длина линии
 - *maxGap* - максимальный промежуток между сегментами линии, лежащими на одной линии, чтобы считать их одним сегментом

Изм.	Лист	№ докум.	Подпись	Дата

- *tolerableSkewAngle* – максимальный допустимый наклон документа
 - *maxLineDistance* - максимальный промежуток между сегментами линии (во время обнаружения клеток)
 - *maxLineGap* - максимальное расстояние между параллельными сегментами, чтобы считать их одним сегментом
 - *lineThickness* – толщина линии
 - *deskew* – если true, то исправить наклон документа
 - *denoise* – если true, то позволить tesseract также применить дополнительные этапы для устранения шума в документе перед распознаванием.
- *Format* – таблица сохраняющая модель (формат) документа. Документ может быть текстовым или содержащим таблицу.
- *name* – название модели
 - *type* – имеет значение ‘TEXT’, если это модель текстового документа, иначе ‘TABLE’
- *TextFormat* – таблица, содержащая модель текстового документа
- *content* – форматированный текст, где переменные указаны по своим названиям и начинаются с буквой ‘\$’
 - *idFormat* – внешний ключ, указывающий на таблицу *Format*.
- *TableFormat* – таблица, содержащая модель документа, из которого нужно найти и извлечь информацию из таблицы.
- *columnCount* – количество столбцов в документе
 - *readFirstLine* – если true, то читать и сохранять первую строку таблицы в качестве данных.
 - *idFormat* – внешний ключ, указывающий на таблицу *Format*.
- *ColumnCharacteristic* – таблица, в которой сохраняются характеристики столбцов читаемого документа.

Изм.	Лист	№ докум.	Подпись	Дата

- *position* – положение этого столбца в таблице
- *name* – название этого столбца
- *type* – тип значения в этом столбце: *INT*, *DOUBLE* или *STRING*
- *idTableFormat* – внешний ключ, указывающий на таблицу *TableFormat*.
- *Project* – таблица, сохраняющая информацию о проектах
 - *name* – название проекта.
 - *inputFilePath* – путь к исходному изображению.

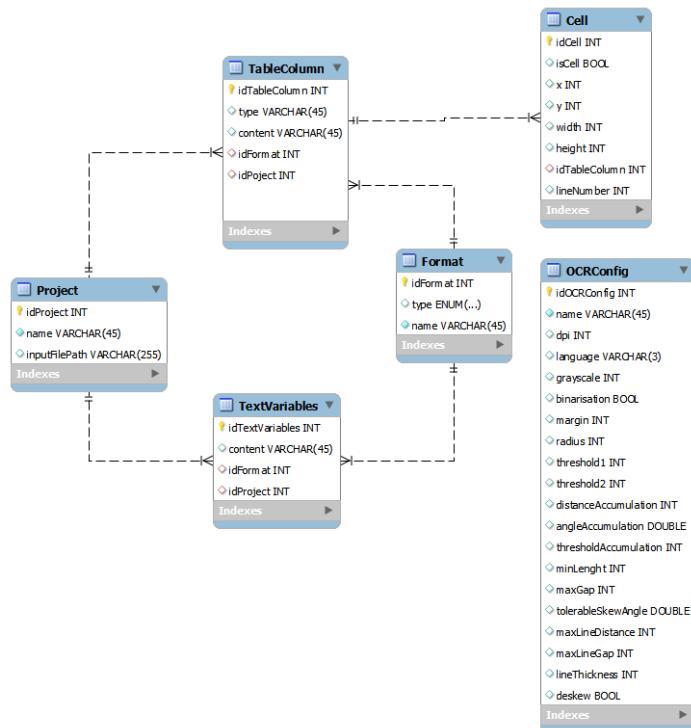


Рисунок 3.1 - ER-диаграмма базы данных этого проекта

3.2.2. Объектно-реляционное отображение с помощью Hibernate

Hibernate — библиотека для языка программирования Java, предназначенная для решения задач объектно-реляционного отображения. Целью Hibernate является освобождение разработчика от значительного объёма сравнительно низкоуровневого программирования по обеспечению хранения объектов в реляционной базе данных. Кроме этого, она не только решает задачу связи классов Java с таблицами базы данных (и типов данных Java с типами данных SQL), но и также предоставляет средства для

Изм.	Лист	№ докум.	Подпись	Дата

автоматической генерации и обновления набора таблиц, построения запросов и обработки полученных данных и может значительно уменьшить время разработки, которое обычно тратится на ручное написание SQL- и JDBC-кода. Hibernate автоматизирует генерацию SQL-запросов и освобождает разработчика от ручной обработки результирующего набора данных и преобразования объектов, максимально облегчая перенос (портирование) приложения на любые базы данных SQL^[23].

3.3. Описание основных классов и методов модуля

3.3.1. Модуль TabulatedOCR

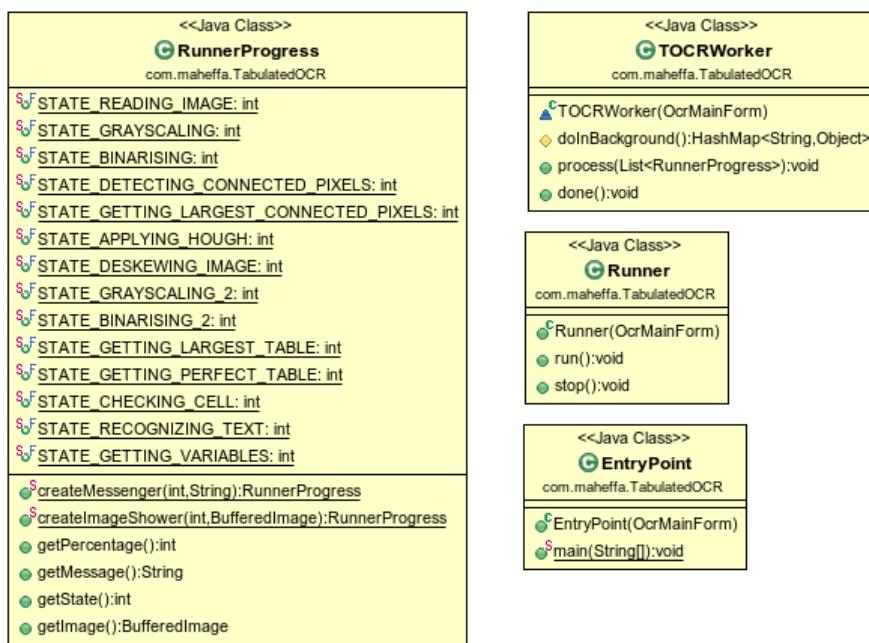


Рисунок 3.2 - Диаграмма классов основного модуля TabulatedOCR

Модуль TabulatedOCR, классы которого на рисунке 3.2, является основным модулем программы, т. е. пункт входа в программу. Здесь осуществляются все шаги программы, начиная с чтения и отчистки изображения и заканчивая сбором и выводом информации.

- Класс *EntryPoint* содержит функцию *main* программы. Конструктору надо передать новую форму *OcrMainForm*, которая является основной графической формой программы. Во время создания нового объекта

Изм.	Лист	№ докум.	Подпись	Дата

EntryPoint инициализируются все элементы графического интерфейса и сессия Hibernate для подключения к базе данных.

- Класс *Runner* — класс для создания отдельного потока, который будет решать задачу, чтобы не тормозить и не мешать работе графического интерфейса.
 - *Void run():* запускает новый поток *TOCRWorker*
 - *void stop ():* остановить работающий поток *TOCRWorker*
- Класс *TOCRWorker* — класс, запущенный в отдельном потоке, выполняющий главную задачу программ
 - *HashMap<String, Object> doInBackground():* Функция, содержащая код основного алгоритма, который будет запущен при нажатии на кнопку «Пуск». Возвращает объект типа словарь, где ‘ключ’ является названием переменной, а ‘значение’ - значением этой переменной в документе.
 - *Void process(List<RunnerProgress> message):* метод, обрабатывающий промежуточную информацию, которую надо сообщить пользователем во время работы алгоритма.
 - *Void done () :* заканчивающий метод, который автоматически вызывается после работы *doInBackground()*. Здесь обрабатывается результат процесса. В этом проекте формирование SQL-скриптов или XML-файлов из прочитанной информации делается именно в этом методе.
- Класс *RunnerProgress* – класс для сообщения, передаваемого изнутри *doInBackground()*, и которое будет обрабатываться параллельно методом *process(message)*.
 - *Static RunnerProgress createMessage (int, String):* создаёт сообщение, содержащее информацию о прогрессе работы (процент и описание)

Изм.	Лист	№ докум.	Подпись	Дата

- Static RunnerProgress createMessage(int, BufferedImage): создаёт сообщение, иллюстрирующее промежуточный рисунок, обработанный на каком-нибудь этапе.

3.3.2. Модуль DBManager

Взаимодействие с базой данных управляется в этом модуле (см. рисунок 3.3). Все классы, кроме *DBAccess*, являются отображениями таблиц базы данных.

Класс *DBAccess* имеет паттерн одиночки. Через экземпляр этого класса осуществляются все записи и считывания базы данных.

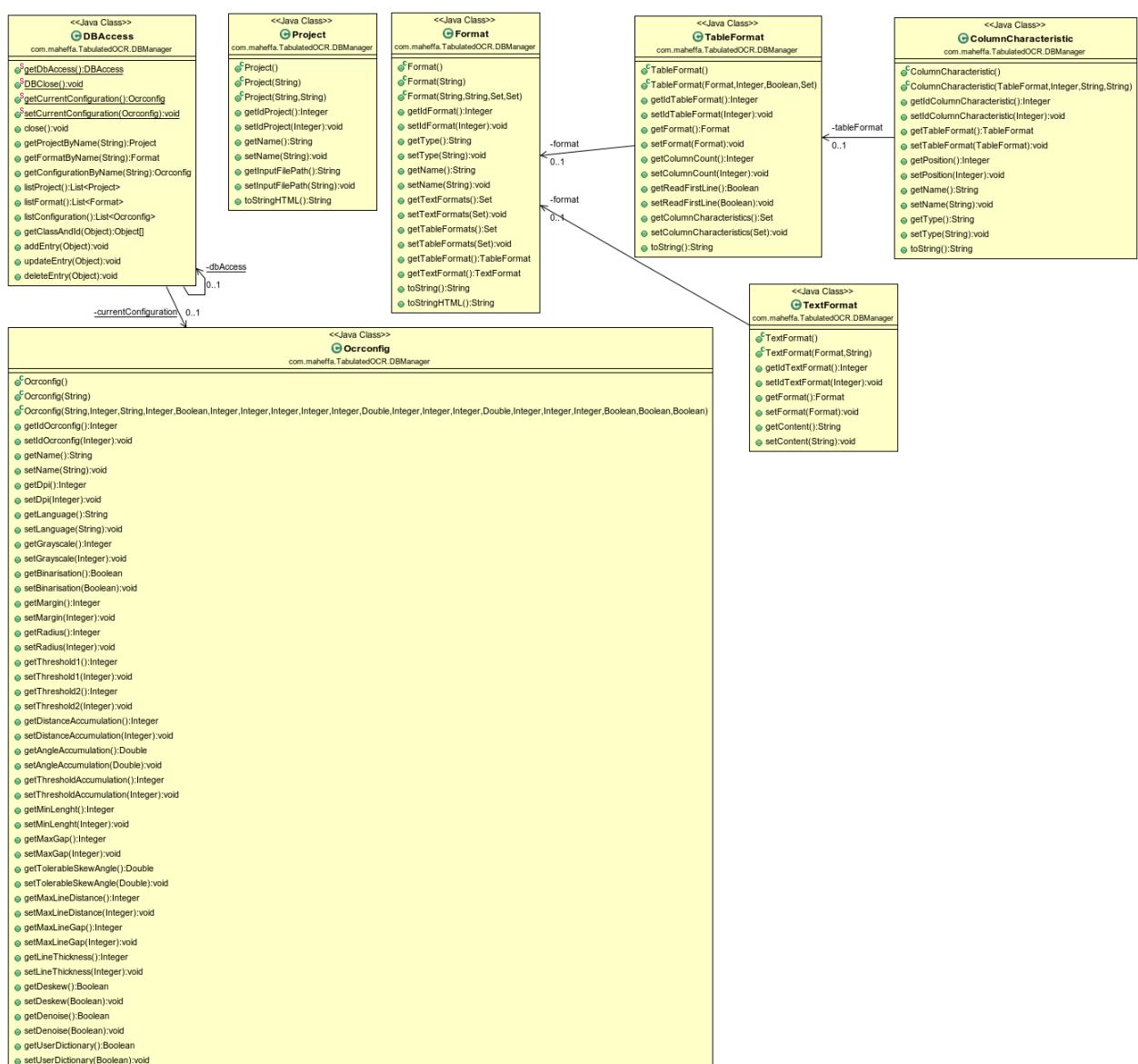


Рисунок 3.3 - Диаграмма классов модуля DBAccess

Изм.	Лист	№ докум.	Подпись	Дата

- *Project getProjectByName (name)*: находит и возвращает проект с названием *name*
- *Format getFormatByName (String name)*: находит и возвращает формат *name*
- *Configuration getConfigurationByName (String name)* : находит и возвращает конфигурацию с названием *name*

3.3.3. Модуль ImageProcessing

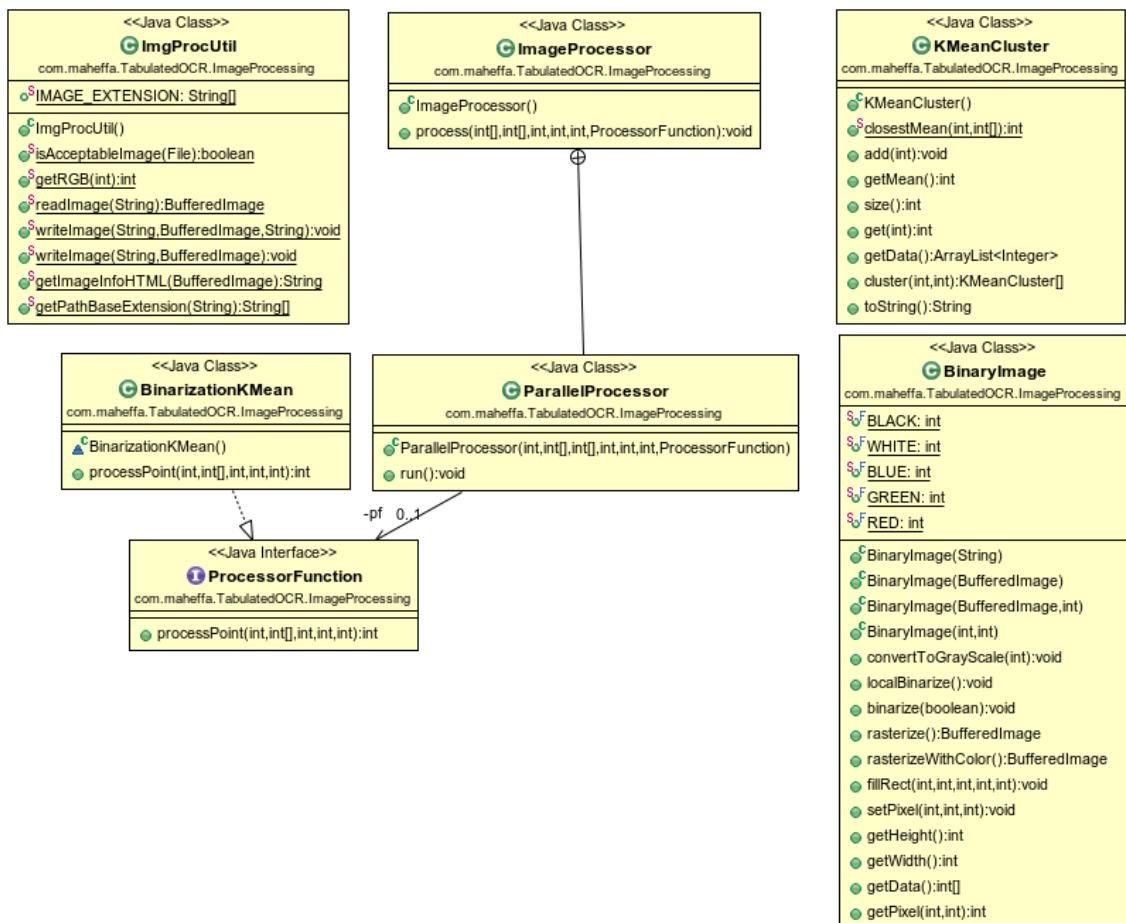


Рисунок 3.4 – Диаграмма классов модуля ImageProcessing

Модуль ImageProcessing, классы которого на рисунке 3.4, осуществляет предварительную обработку изображения без использования внешних библиотек.

- Класс *BinaryImage*: бинарное изображение. Конструктор принимает в качестве параметра цветное изображение.

Изм.	Лист	№ докум.	Подпись	Дата

- `Void convertToGrayScale(int method)` : преобразовать в чёрно-белое, используя метод `method`
- `Void binarize()` : бинаризовать изображение
- Класс `KMeansCluster`: класс кластер
 - `KmeansCluster[] cluster(numCluster, maxIter)` : кластеризирует кластер на `numCluster` кластеров, итерация кластеризации не должна превышать `maxIter`
 - `Int getMean()` : возвращает центроид этого кластера
 - `Void add(value)` : добавляет `value` в этот кластер
- Класс `ProcessFunction` является интерфейсом для функции обработки изображения, которая обрабатывает пиксели изображения по отдельности. Такие функции (приведение в чёрно-белое, фильтрация, и т.д.) можно параллелизовать.
- Класс `ImageProcessor` позволяет распараллелизовать работы некоторой функции над одним изображением.
 - `void process(int[] src, int[] dst, int height, int width, int area, ProcessorFunction pf)` : Запускает функции `pf` над пикселям из `src` и записывает результат в `dst`. `Height` и `width` – размер изображения. `area` – локальная площадь, с которой эта функция должна работать.

3.3.4. Модуль TableStructureDetection

Модуль TableStructureDetection (см. рисунок 3.5) для обнаружения наклона изображения и структуры таблицы, а также нахождения всех возможных клеток.

- Класс `TableDetector` – основной класс этого модуля, через который осуществляются все операции с обнаружением наклона и остальными обработками отсканированного документа.

Изм.	Лист	№ докум.	Подпись	Дата

- static BufferedImage rotate(img, angle) : вращает рисунок img вокруг его центра на угол angle. Используется для корректировки наклона. Возвращает исправленный рисунок

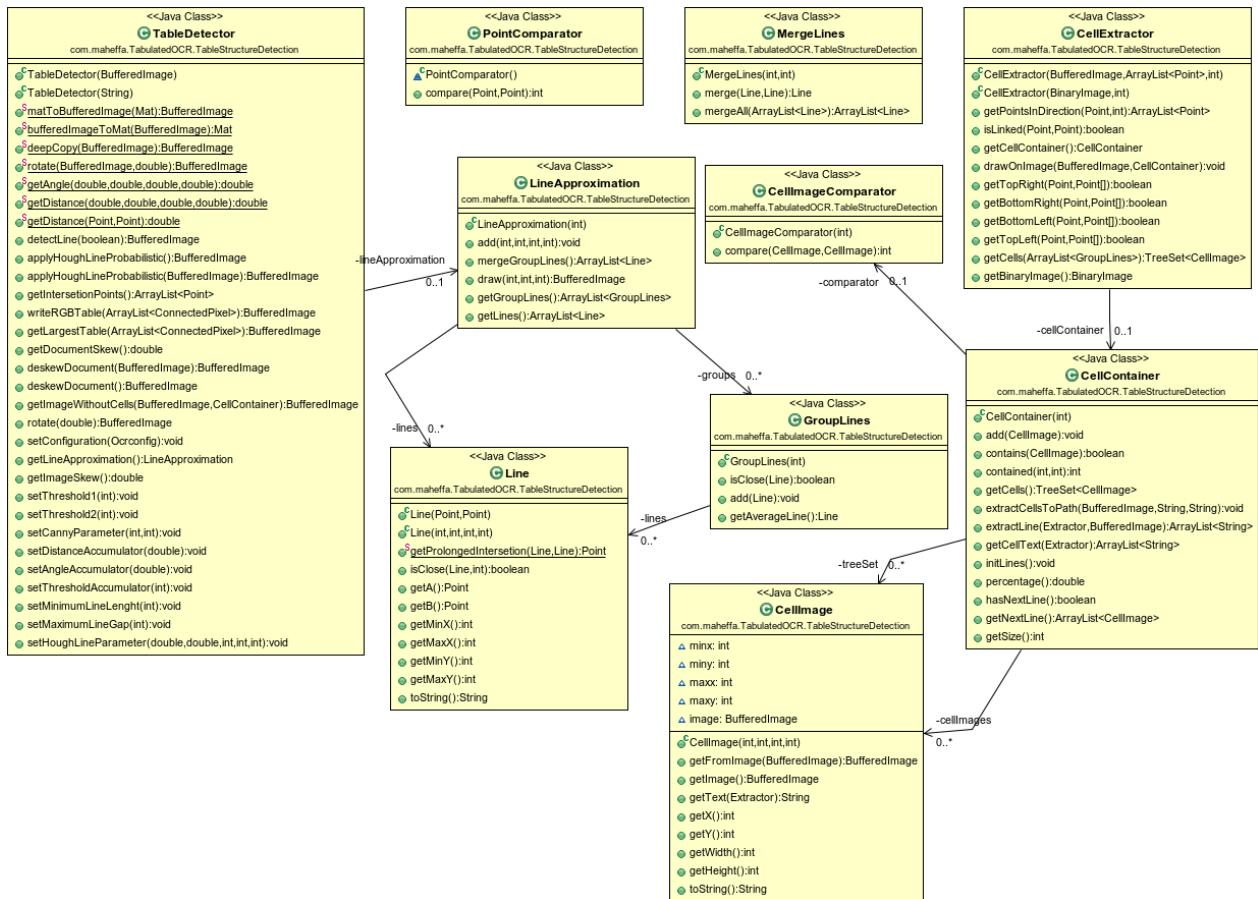


Рисунок 3.5 – Диаграмма классов модуля TableStructureDetection

- BufferedImage applyHoughLineProbabilistic (BufferedImage image) : принимает фильтрации по оператору Кэнни и преобразование Хафа над входным изображением image. На выходном изображение отображается все обнаруженные отрезки. Побочным эффектом этой функции является сохранение координаты обнаруженных отрезков. Эти координаты необходимы для определения наклона документа.
- ArrayList<Point> getIntersectionPoints() : возвращает все точки пересечения всех обнаруженных отрезков.
- double getDocumentSkew() : рассчитывает наклон документа.

Изм.	Лист	№ докум.	Подпись	Дата

- `void setConfiguration(Ocrconfig conf)` : устанавливает значение параметров текущей конфигурации, полученных из базы данных.
- Класс `CellImage` – представляет собой клетку в изображении. Он содержит координаты (`x, y`) и размеры (`width, height`) одной клетки.
 - `BufferedImage getFromImage (BufferedImage img)`: возвращает кусочек изображения, представлявший собой клетку, по координатам и размерам, указанным в этом классе.
- Класс `CellContainer` – представляет собой набор клеток `CellImage` одной таблицы.
 - `void add(CellImage e)` : добавляет клетку `e` в набор, если её еще в нем нет.
 - `void initLines()` : метод, инициализирующий процесс считывания текстов по строке таблицы сверху вниз.
 - `boolean hasNextLine()` : возвращает `true`, пока последняя прочитанная строка не является последней в таблице.
 - `ArrayList<CellImage> getNextLine()` : возвращает список клеток на следующую строку.
 - `ArrayList<String> extractLine(Extractor extractor, BufferedImage original)` : извлекает список текстов из клеток, находившихся в одной строке таблицы. `Extractor` – объект, с помощью которого можно распознавать тексты. `Original` – изображение, из которого нужно извлечь тексты.
- Класс `CellImageComparator` – сравнитель порядка клетки. Конструктору этого класса передаётся пороговое значение, которое помогает определить, если две клетки находятся в одной строке. Клетки на верхней части изображения имеют больше приоритета, чем клетки в нижней части. Если клетки в одной строке, то приоритет левой клетки имеет большую приоритетность

Изм.	Лист	№ докум.	Подпись	Дата

- `int compare(CellImage c1, CellImage c2)` : возвращает 0, отрицательное или положительное значение, если приоритет клетки `c1` равен, больше или меньше приоритета, чем `c2` соответственно.
- Класс `CellExtractor` – извлекает клетки и информацию в них. Конструктору необходимо передать изображение и найденные точки пересечения.
 - `ArrayList<Point> getPointsInDirection(Point p, int direction)` : получить список точек пересечении, которые находятся вверху (`direction = 0`), справа (`direction = 1`), внизу (`direction = 2`), слева (`direction = 3`) точки `p`. Этот список будет нужен в процессе, описанном в параграфе *0*, при нахождении четырех составных точек клетки, верхний левый уголок которой является точкой `p`.
 - `CellContainer getCellContainer()` : запускает процесс обнаружения всех клеток и возвращает найденные клетки в контейнере.
- Класс `Line` – отрезок, конструктору которого необходимо передать два краевых пункта.
 - `static Point getProlongedIntersection(Line l1, Line l2)` : возвращает точку пересечения двух отрезков.
- Класс `GroupLines` – представляет собой набор всех отрезков, находящихся достаточно близко друг к другу, которые возможно слить в один отрезок.
 - `boolean isClose(Line l)` : возвращает `true`, если отрезок `l` близок к любым отрезкам этого набора.
 - `Line getAverageLine()` : возвращает либо горизонтальный, либо вертикальный отрезок, края которого являются среднем значением соответствующих краев каждого отрезка этого набора.
- Класс `LineApproximation` организует обнаруженные отрезки, полученные от преобразования Хафа, и определяет форму таблицы.

Изм.	Лист	№ докум.	Подпись	Дата

- `void add(int x1, int y1, int x2, int y2)`: добавляет координаты отрезка, найденного по преобразованию Хафа, и определяет, в каком наборе GroupLines он должен входить. Если нет такого набора, то создаётся новый набор.
- `ArrayList<Line> mergeGroupLines()` : метод, в котором осуществляется процедура слияния линии, описанная в параграфе 0.

3.3.5. Модуль TextExtraction

Модуль TextExtraction (см. рисунок 3.6) для распознавания текстов.

- Класс `ConnectedPixel` представляет собой набор смежных пикселей.
 - `static ArrayList<ConnectedPixel> getConnectedPixels(int radius, int margin, BinaryImage img)`: функция, которая возвращает список наборов смежных чёрных пикселей, находящихся в изображении `img`. Два пикселя считаются смежными, если расстояние между ними меньше `radius`, а также два набора считаются одним набором, если расстояние между ними меньше `margin`

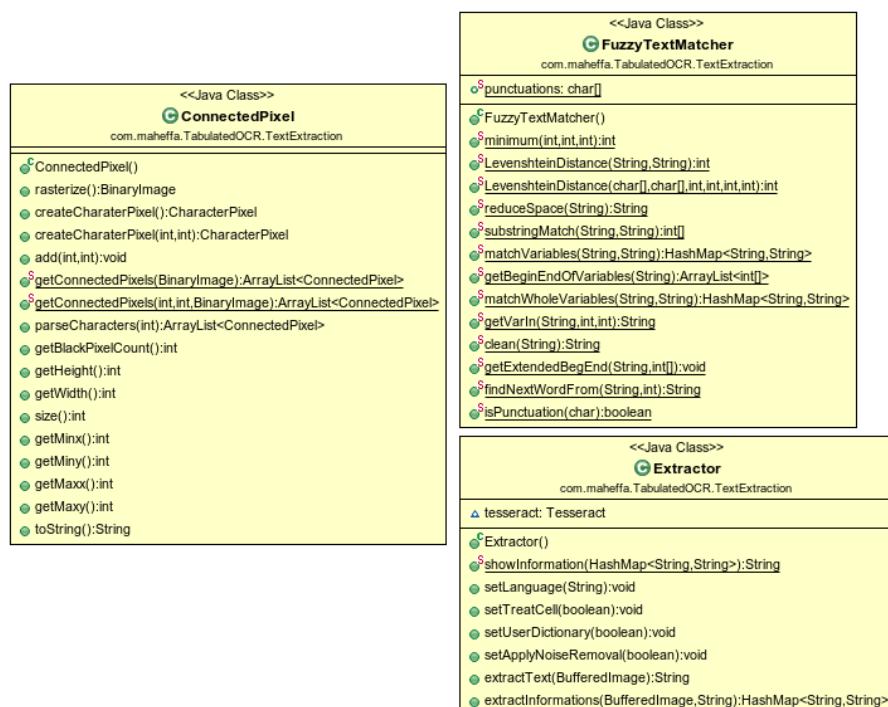


Рисунок 3.6 – Диаграмма классов модуля *TextExtraction*

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

- Класс *Extractor* содержит метод для распознавания текстов
 - *void setLanguage(String lang)* : устанавливает язык документа
 - *String extractText(BufferedImage img)* : распознает все тексты в изображении *img*
- Класс *FuzzyTextMatcher* содержит методы для управления этапами для извлечения значения переменных по шаблону.
 - *static int LevenshteinDistance(char[] str1, char[] str2, int i0, int j0, int i1, int j1)*: определяет расстояние Левенштейна между подстроками *str1[i0-j0]* и *str2[i1-j1]*
 - *static HashMap<String, String> matchWholeVariables(String text, String format)*: извлекает значение переменных из распознанного текста *text* по шаблону *format*. Возвращает множество пар *<ключ, значение>*, где ключ есть название переменной и значение как значение этой переменной
 -

Изм.	Лист	№ докум.	Подпись	Дата

4. Вычислительный эксперимент

Для доказательства работы программы проводится 2 эксперимента по каждому из двух шаблона: форматированный текстовой документ и документ с таблицей определённой структуры.

4.1. Вычислительный эксперимент 1: форматированный текстовой документ

4.1.1. Входные данные

В качестве входных данных был выбран отсканированный документ, такой как упрощённая версия регистрационной карты из ФМС России, который показан на рисунке 4.1.

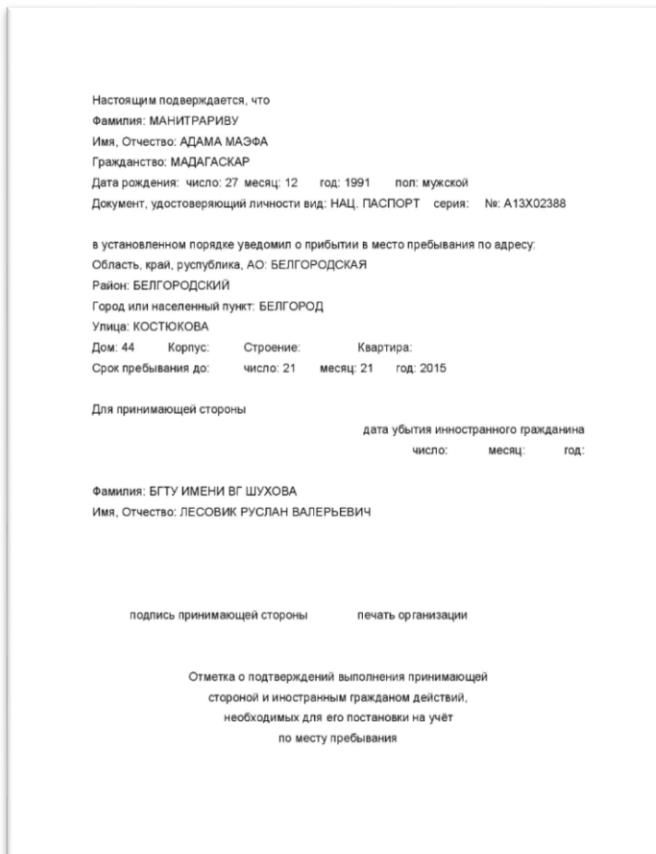


Рисунок 4.1 - Скан регистрационной карты иностранного студента
БГТУ им Шухова

Изм.	Лист	№ докум.	Подпись	Дат	ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ		
Разраб.	Манитрапишу А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	Лит.	Лист
Руковод.	Брусенцев А.И.					61	Листов
Консул.						116	
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.						

Программе также нужно передать формат документа (см. рисунок 4.2).

В этом случае программе передан следующий форматированный текст в качестве шаблона документа:

```

Настоящим подтверждается, что
Фамилия: $lastname
Имя, Отчество: $firstname
Гражданство: $country
Дата рождения: число: $dbirth месяц: $mbirth год: $ybirth пол:
$sex
Документ, удостоверяющий личности вид: $document серия: $serie
№: $number
в установленном порядке уведомил о прибытии в место пребывания
по адресу:
Область, край, республика, АО: $region
Район: $area
Город или населенный пункт: $city
Улица: $street
Дом: $house Корпус: $corpus Строение: $building Квартира:
$appartement
Срок пребывания до: число: $dend месяц: $mend год: $yend
Для принимающей стороны дата убытия иностранного гражданина
число:           месяц:           год:
Фамилия: $hostfirstname
Имя, Отчество: $hostlastname
подпись принимающей стороны      печать организации
Отметка о подтверждении выполнения принимающей
стороной и иностранным гражданином действий,
необходимых для его постановки на учёт
по месту пребывания

```

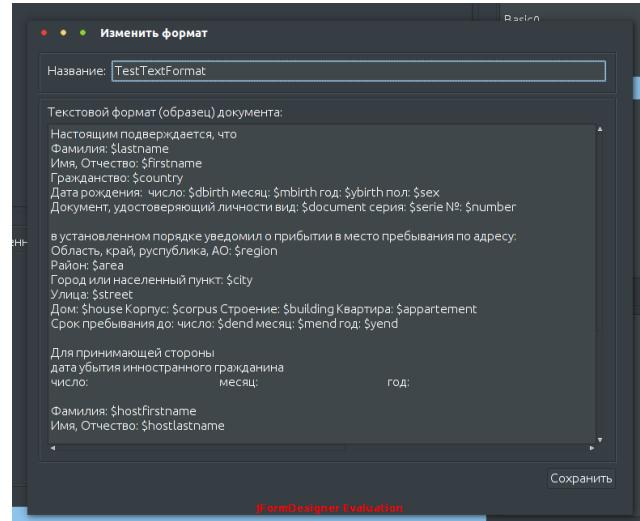


Рисунок 4.2 – Передача шаблон программы

Изм.	Лист	№ докум.	Подпись	Дата

Название переменных в форматированном тексте должно быть задано последовательностью из латинских букв, цифр и знаков ‘_’ и должно начинаться со знака ‘\$’. Программа автоматически находит все расположения переменных с помощью поиска по регулярному выражению. Значение этих переменных программа находит в распознанном тексте и извлекает в XML-файл.

Как задать формат будет рассмотрено в следующей главе.

4.1.2. Выходные данные

Перед распознаванием текста, при создании шаблона форматированного текста, необходимо найти наклон документа и исправить его для улучшения результата распознавания.

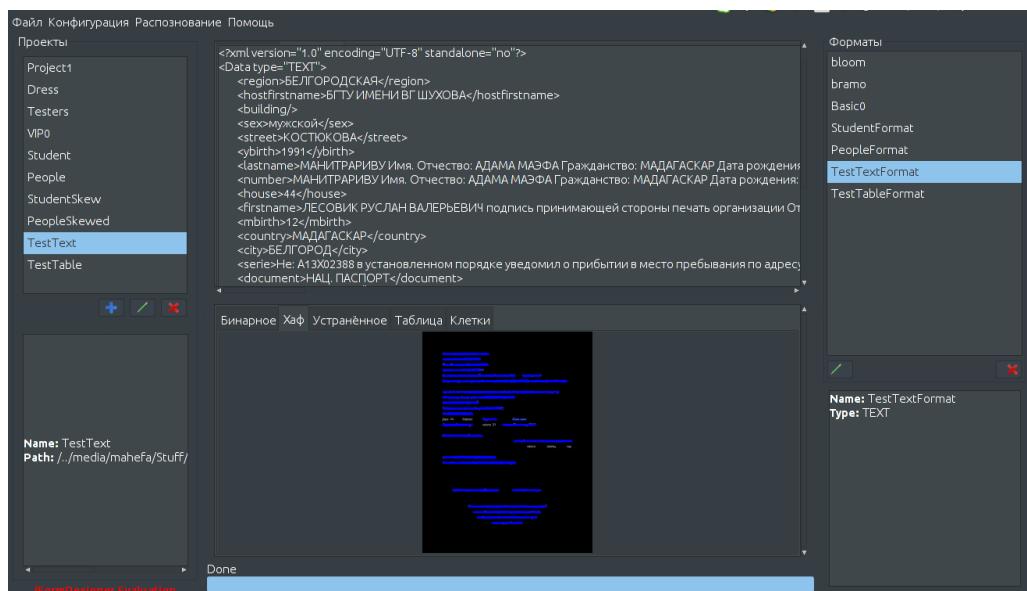


Рисунок 4.3 – Исключение ключевой информации из скана упрощённой регистрационной карты иностранного студента БГТУ им Шухова

Как изображено на рисунке 4.3, результат распознавания описан в XML-формате, а результата преобразования Хафа в окне с названием “Хаф”.

4.1.3. Анализ результата

Как видно на рисунке 4.4, не все переменные были правильно исключены. Это объясняется тем, что библиотека Tesseract читает текст и переводит его в одну строку. Это естественно усложняет нахождение конца строки исходного документа. Из-за этого в значение переменной расположено

Изм.	Лист	№ докум.	Подпись	Дата

значение нескольких строк. При использование лучшей библиотеки, например, платной Abbyy SDK, результат распознавания может быть лучше и значения переменных могут быть без ошибок распознаны.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<data type="TEXT">
<region>БЕЛГОРОДСКАЯ</region>
<hostfirsname>БЕТУ ИМЕНИ ВИ ШУХОВА</hostfirsname>
<buildIngr/>
<sex>мужской</sex>
<street>КОСТЯКОВА</street>
<birth>1991</birth>
<lastname>МАНГАРАИВА</lastname>
<number>АДАМА МАЭФА Гражданство: МАДАГАСКАР Дата рождения: число: 27 месяц: 12 год: 1991 пол: мужской</number>
Документ, удостоверяющий личности вид: НАЦ. ПАСПОРТ серия: №: А13Х02388 в установленном порядке уведомил о прибытии в место пребывания по адресу: Область, край, республика, АО: БЕЛГОРОДСКАЯ Район: БЕЛГОРОДСКИЙ город или населенный пункт: БЕЛГОРОД Улица: КОСТЯКОВА Дом: 44 Корпус: Строение: Квартира: Срок пребывания до: число: 21 месяц: 21 год: 2015 Для принимающей стороны дата убытия иностранного гражданина число: мес: год: Фамилия: БЕТУ ИМЕНИ ВИ ШУХОВА Имя, Отчество: АДАМА МАЭФА Гражданство: МАДАГАСКАР Дата рождения: число: 27 месяц: 12 год: 1991 пол: мужской</number>
<firstname>ЛЕСОВИК РУСЛАН ВАЛЕРЬЕВИЧ подпись принимающей стороны печать организации Отметка о подтверждении выполнения принимающей стороны и иностранного гражданина действий, необходимых для его постановки на учет по месту пребывания</firstname>
<country>МАДАГАСКАР</country>
<city>БЕЛГОРОД</city>
<series>№: А13Х02388 в установленном порядке уведомил о прибытии в место пребывания по адресу: Область, край, республика, АО: БЕЛГОРОДСКАЯ Район: БЕЛГОРОД город или населенный пункт: БЕЛГОРОД Улица: КОСТЯКОВА Дом: 44 Корпус: Строение: Квартира: Срок пребывания до: число: 21 месяц: 21 год: 2015 Для принимающей стороны дата убытия иностранного гражданина число: мес: год: Фамилия: БЕТУ ИМЕНИ ВИ ШУХОВА Имя, Отчество: ЛЕСОВИК РУСЛАН ВАЛЕРЬЕВИЧ подпись принимающей стороны печать организации Отметка о подтверждении выполнения принимающей стороны и иностранного гражданина действий, необходимых для его постановки на учет по месту пребывания</series>
<address>НАЦ. ПАСПОРТ</address>
<ageas></ageas>
<apartmenttype></apartmenttype>
<end>21</end>
<rend>12</rend>
<yend>1991</yend>
<Document>Документ, удостоверяющий личности вид: НАЦ. ПАСПОРТ серия: №: А13Х02388 в установленном порядке уведомил о
```

Рисунок 4.4 – результирующий XML-файл

4.2. Вычислительный эксперимент 2: документ с таблицей определённой структуры

4.2.1. Входные данные

Для этого шаблона, в качестве входных данных был выбран документ из деканата кафедры ПОВТиАС института ИИТУС университета БГТУ им Шухова.

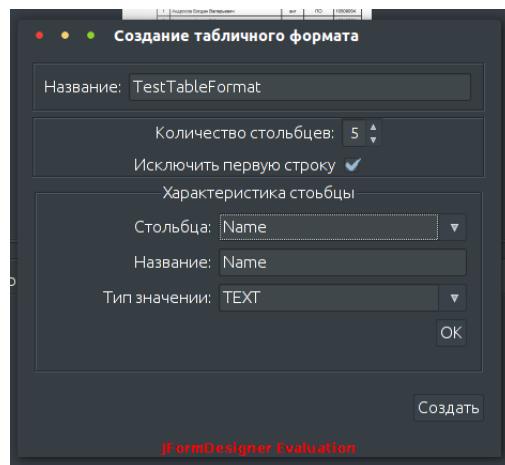


Рисунок 4.5 – указание структуры таблицы, содержащей список студентов

Изм.	Лист	№ докум.	Подпись	Дата

В этом документе есть список студентов группы ПВ-51, иностранных языков, которые они изучают, формы обучения (бюджетник или платная основа), и номер зачётной книжки. Этот документ отображён на рисунке 4.6.

4.2.2. Выходные данные

Белгородский государственный технологический университет им. В. Г. Шухова				
группа: ПВ-51 Специальность: 230105-03				
Институт: ИИТУС Куратор: Староста: Обухов Артём Юрьевич				
№	Фамилия Имя Отчество	ин. яз.	Осново обучения	№ зач. книжки
1	Андросов Богдан Валерьевич	анг	ПО	10509054
2	Анищенко Алексей Игоревич	анг		105105081
3	Воробьёв Роман Викторович	анг		105105105
4	Жадан Антон Павлович	анг		105105106
5	Картамышев Сергей Владимирович	анг		105105108
6	Коваленко Татьяна Владимировна	анг		105105109
7	Лебедев Денис Константинович	анг		105105110
8	Линев Андрей Александрович	анг		105105088
9	Литвинова Юлия Александровна	анг		105105112
10	Манилариву Адама Маэфа	фра	ВО бюджет	105105163
11	Обухов Артём Юрьевич	анг		105105092
12	Силина Татьяна Сергеевна	анг		105105117
13	Сипачев Антон Игоревич	анг		105105097
14	Ткаченко Александр Юрьевич	анг		105105122
15	Трунов Юрий Анатольевич	анг		105105124
16	Чеува Нгонг Микаэль Гээль	анг	ВО бюджет	105105165
17	Шахов Максим Владимирович	анг		105105102
18	Штанов Александр Евгеньевич	анг		105105127
19	Юсюмбели Николай Иванович	нем	ВО бюджет	105105161

Директор института _____ / Рубанов Василий Григорьевич /

Рисунок 4.6 – список студентов группы ПВ-51 (2014-2015), факультета ИИТУС, университета БГТУ им Шухова

Найдение наилучших параметров для правильного обнаружения таблицы нетривиально, так как проходит через несколько этапов. Был

Изм.	Лист	№ докум.	Подпись	Дата	Лист	65
					ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ	

проводен эксперимент с разными параметрами, ниже показаны результаты двух разных конфигурации: первая конфигурация – конфигурация по умолчанию, вторая – конфигурация с хорошим результатом (см. рисунок 4.7).

По этих конфигурациях, результат промежуточных преобразований изображение показан на рисунке 4.8.

В результате обработки по первой конфигурации были распознаны только 4 строки, но даже их значения были неправильны. По второй конфигурации, были распознаны все строки.

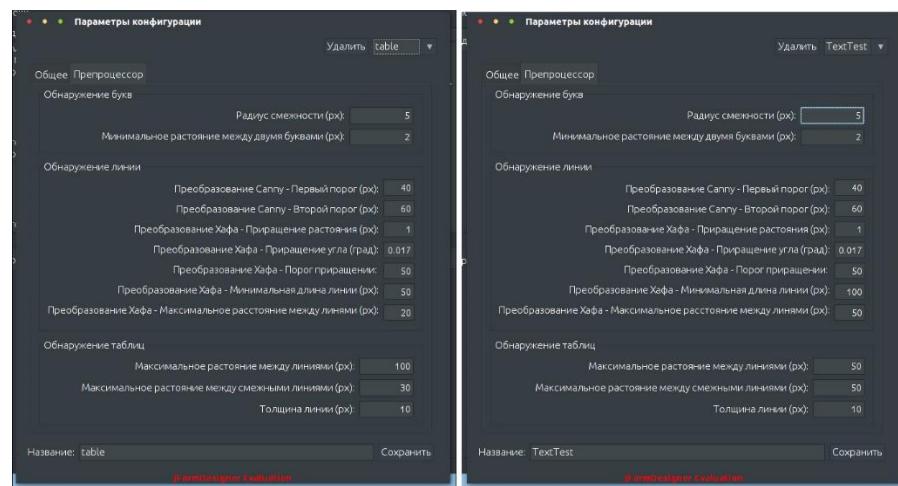


Рисунок 4.7 – Две разных конфигурации для обнаружения таблицы



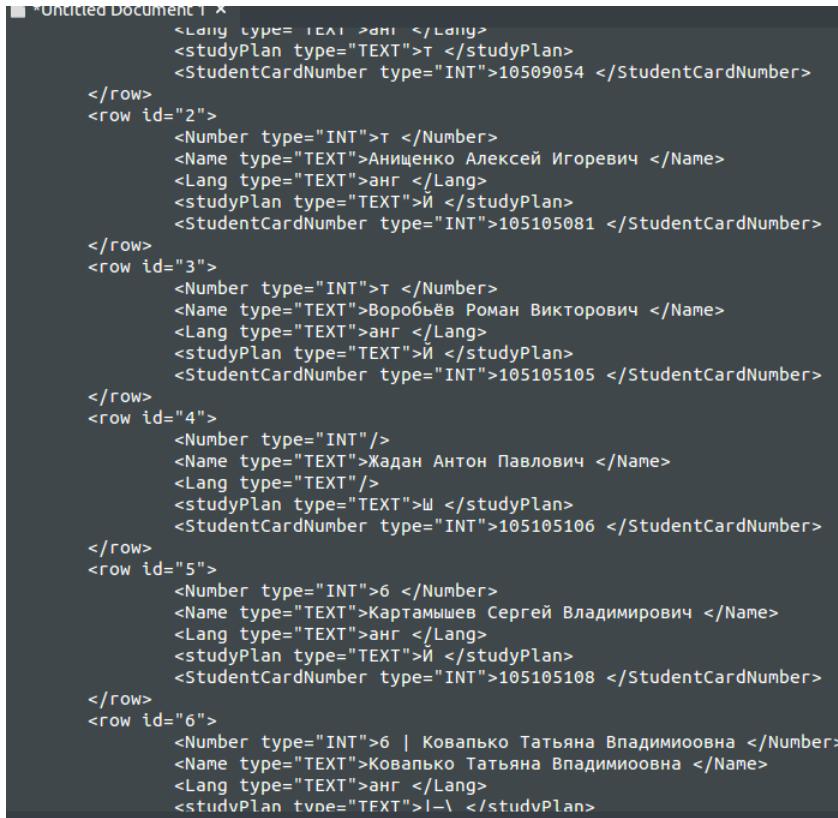
Рисунок 4.8 – Применение преобразования Хафа, выделения и слияния ребер таблицы и обнаружения всех клеток под 2м разными конфигурациями.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

Хотя значения в первом столбце Tesseract принимал как буква, вместо числа, большинство значений в остальных столбцах было удачно распознано.

Результирующий XML-файл на рисунке 4.9 имеет следующий формат:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Data type="TABLE">
    <row id="1">
        <Number type="INT">1</Number>
        <Name type="TEXT">Андрюсов Богдан Вальеревич</Name>
        <Lang type="TEXT">анг</Lang>
        <StudyPlan type="TEXT"/>
        <StudentCardNumber type="INT"> 10509054 </StudentCardNumber>
    </row>
</Data>
```



```
*Untitled Document 1
<Number type="INT">1</Number>
<Name type="TEXT">Андрюсов Богдан Вальеревич</Name>
<Lang type="TEXT">анг</Lang>
<StudyPlan type="TEXT"/>
<StudentCardNumber type="INT"> 10509054 </StudentCardNumber>

</row>
<row id="2">
    <Number type="INT">2</Number>
    <Name type="TEXT">Анищенко Алексей Игоревич </Name>
    <Lang type="TEXT">анг </Lang>
    <studyPlan type="TEXT">И </studyPlan>
    <StudentCardNumber type="INT">105105081 </StudentCardNumber>
</row>
<row id="3">
    <Number type="INT">3</Number>
    <Name type="TEXT">Воробьев Роман Викторович </Name>
    <Lang type="TEXT">анг </Lang>
    <studyPlan type="TEXT">И </studyPlan>
    <StudentCardNumber type="INT">105105105 </StudentCardNumber>
</row>
<row id="4">
    <Number type="INT"/>
    <Name type="TEXT">Хадан Антон Павлович </Name>
    <Lang type="TEXT"/>
    <studyPlan type="TEXT">Ш </studyPlan>
    <StudentCardNumber type="INT">105105106 </StudentCardNumber>
</row>
<row id="5">
    <Number type="INT">6</Number>
    <Name type="TEXT">Картамышев Сергей Владимирович </Name>
    <Lang type="TEXT">анг </Lang>
    <studyPlan type="TEXT">И </studyPlan>
    <StudentCardNumber type="INT">105105108 </StudentCardNumber>
</row>
<row id="6">
    <Number type="INT">6 | Коваленко Татьяна Владимировна </Number>
    <Name type="TEXT">Коваленко Татьяна Владимировна </Name>
    <Lang type="TEXT">анг </Lang>
    <studyPlan type="TEXT">І-1 </studyPlan>
```

Рисунок 4.9 – Результат чтения таблицы в изображении

Где Number, Name, Lang, StudyPlan и StudentCardNumber означает номер, ФИО, изучаемый иностранный язык, форма обучения и номер зачётной книжки студента, соответственно. Они должны соответствовать столбцам таблицы документа и быть заданы как шаблон, который программа использует для извлечения этой информации.

Изм.	Лист	№ докум.	Подпись	Дата

4.2.3. Анализ результата

При правильной настройке программы можно получить хороший результат извлечения информации с корректным количеством строк в XML-файле. Так как документы одного типа могут иметь одинаковые качества и образы, программа представляет возможность сохранения специфической настройки. Тогда пользователю просто необходимо загрузить ту подходящую настройку для данного документа.

Вывод

После проведения экспериментов было показано, что программа обеспечивает обнаружение форматированного текста и структуры таблицы, обнаружение текста и выделение ключевой информации.

Недостатком является несовершенство распознавания текста, а также извлечение значений некоторых переменных. Успешность распознавания текста зависит от качества изображения, предобработки изображения и качества использованной библиотеки. Замечается также, что для каждого типа документа необходимо настраивать приложение по-разному.

Изм.	Лист	№ докум.	Подпись	Дата

5. Руководство пользователя

5.1. Введение

При запуске приложения открывается общее окно, представляющее собой пользовательский интерфейс, как показано на рисунке 5.1.

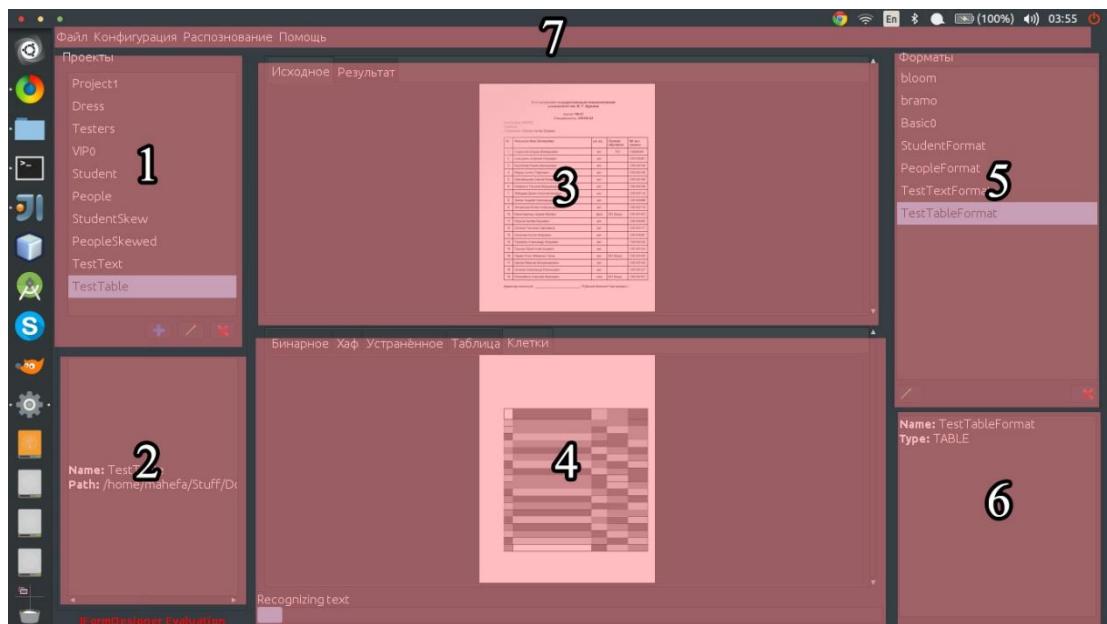


Рисунок 5.1 – Пользовательский интерфейс

Интерфейс разделяется на 7 основных частей:

- 1. Раздел проекта:** здесь находятся список названий всех существующих проектов в базе данных и кнопки для создания, редактирования и удаления проекта из базы данных.
 - 2. Раздел описания проекта:** где описывается проект, выбранный в разделе проекта.
 - 3. Раздел входных и выходных данных:** в этом разделе заключается два окна: окно, показывающее входное изображение, соответствующее выбранному проекту, и окно, в котором записывается результат работы программы перед их сохранением в файл.

Изм.	Лист	№ докум.	Подпись	Дат	РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ		
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей		
Руковод.	Брусенцев А.И.				Лит.	Лист	
Консул.						Листов	
Н. контр.	Полунин А.И.					69	
Зав. каф.	Поляков В.М.					116	
					БГТУ им. В. Г. Шухова, ПВ-51		

4. **Раздел промежуточных результатов:** где показывается развитие процесса решения задачи. Он располагается в 5 окне, и содержит результаты: бинаризации, преобразования Хафа, документ без наклона, обнаруженные ребра таблицы и обнаруженные клетки.
5. **Раздел шаблонов:** здесь указывается список всех шаблонов, находящихся в базе данных.
6. **Раздел описания шаблонов:** где описывается шаблон, выбранный в разделе 5.
7. **Меню:** главное меню программы.

5.2. Настройка

Чтобы получить наилучший результат, необходимо настроить приложение. Чтобы открыть окно настройки, выбираем в **меню** → **Конфигурация** → **Параметры распознавания** (см. рисунок 5.2)

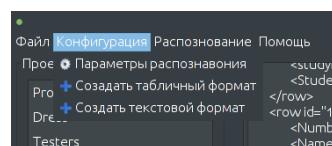


Рисунок 5.2 – Открывание окна конфигурации

Появляется окно, как показано на рисунке 5.3.

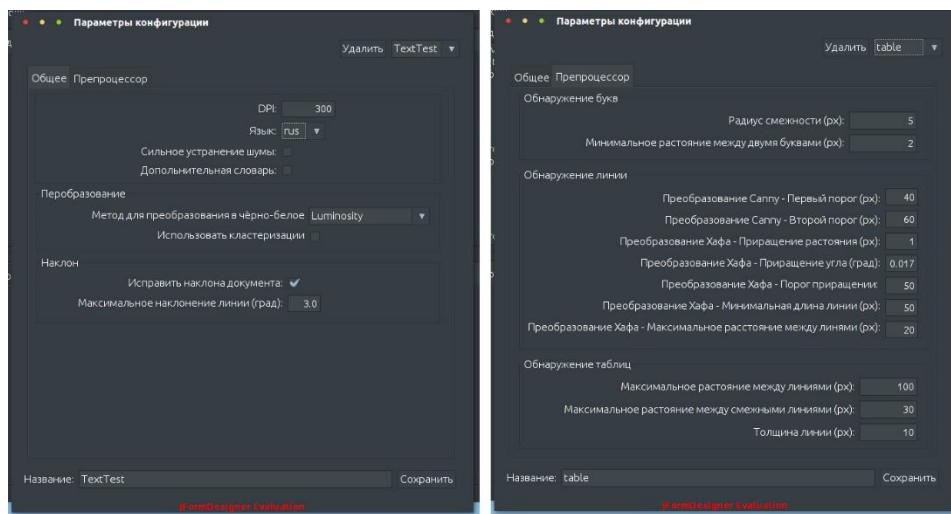


Рисунок 5.3 – Параметры распознавания

Конфигурацию можно сохранить (кнопкой «сохранить»), удалить (кнопкой «удалить»). Можно выбрать какую конфигурации из сохранённых в

Изм.	Лист	№ докум.	Подпись	Дата

базе данных нужно использовать, выбирая одну из конфигурации в выпадающем списке рядом с кнопкой «удалить».

Описание параметров:

- **DPI**: разрешение, с которым приложение будет работать. Если разрешение изображения не совпадает с этим числом, то перед обработкой изображение сначала конвертируется.
- **Язык**: язык документа. Для выбора есть 3 языка: русский (rus), французский (fra) и английский (eng).
- **Дополнительная словарь**: если выбрано, то для распознавания используется дополнительный пользовательский словарь.
- **Метод преобразования в чёрно-белое**: выбор одного из методов, описанных в главе 2.1.1.
- **Использовать кластеризации**: если выбран, то используется метод K-Means для бинаризации изображения, иначе выбирается порог бинаризации 128.
- **Исправить наклон документа**: если не выбран, то программа пропускает проверку и исправление наклона документа.

Все параметры в окне «препроцессор» не требуют пояснений, так как их значения уже хорошо описаны. Эти значения являются входными параметрами алгоритмов Кэнни, преобразования Хафа, обнаружения смежных пикселей и обнаружения структуры таблицы.

5.3. Проект

Чтобы создать проект, в разделе проекта нужно щелкнуть на кнопку со значком “+”. Для изменения и удаления надо щелкнуть соответственно на вторую и третью кнопку, как показано на рисунке 5.4.

При нажатии, по кнопке создания проекта появится окно, которое изображено на рисунке 5.5. В этом окне нужно указать название проекта и путь к исходному файлу.

Изм.	Лист	№ докум.	Подпись	Дата

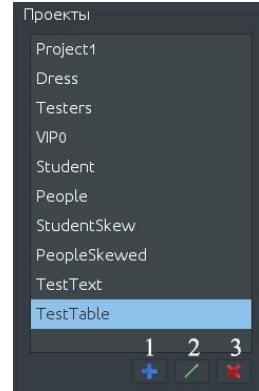


Рисунок 5.4 – создание, редактирование и удаление проекта

При нажатии на кнопку выбрать открывается эксперт для выбора файла в системе. Название проекта не должно совпадать с уже существующим названием проекта.

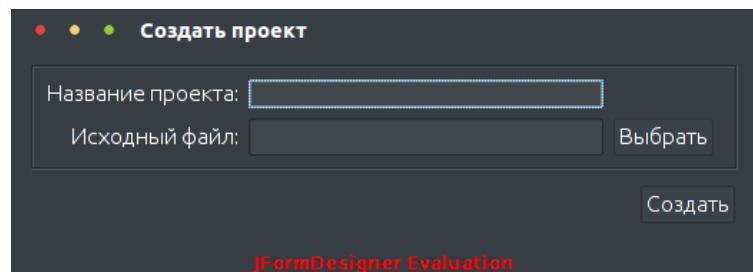


Рисунок 5.5 – Окно создания проекта

5.4. Шаблоны

Чтобы создать шаблон, нужно выбрать в меню → конфигурация как указано на рисунке 5.6.

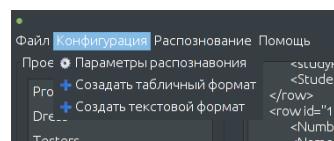


Рисунок 5.6 – Путь к созданию шаблона

Есть два возможных типа шаблона: табличный формат и текстовый формат.

5.4.1. Табличный формат

При выборе создания шаблона табличного формата открывается окно как показано на рисунке 5.7.

Нужно указать название шаблона. Это название не должно совпадать с уже существующим название шаблона. Также нужно указать количества

Изм.	Лист	№ докум.	Подпись	Дата

столбцов. В выпадающем списке будут соответствующее количество элементов. Необходимо выбрать каждый элемент (столбец) последовательно и указать его название и тип значения. Затем нажать на кнопку **OK** и перейти к следующему столбцу.

Если галочка «исключить первую строку» поставлена, то первую строку таблицу не читается, так как первая строка обычно является названием столбцов.

Чтобы сохранить шаблон, нажать на кнопку «Создать».

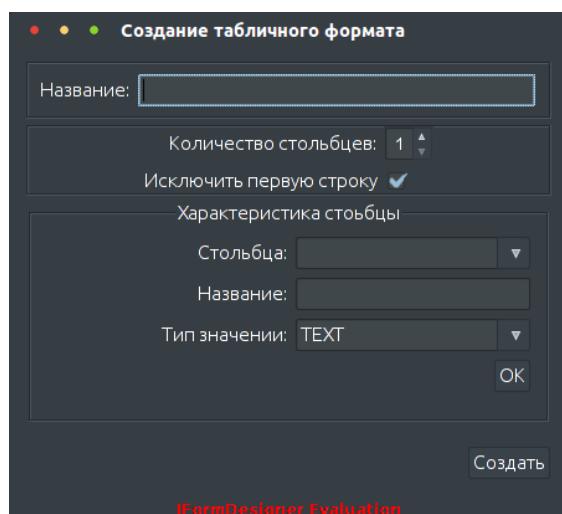


Рисунок 5.7 – Создание шаблона табличного формата

5.4.2. Текстовой формат

При выборе создания шаблона форматированного текста открывается окно как показано на рисунке 5.8.

Нужно указать название шаблона. Это название не должно совпадать с уже существующим название шаблона. В разделе «текстовой формат (образец)» необходимо описать образец документа форматированным текстом.

Форматированный текст должен указывать константы и переменные. Под константами понимается те части документа, которые остаются одинаковыми для разных документов одного типа. Под переменными понимаются те части документа, которые отличают один документ от другого. Название переменных в форматированном тексте должно быть задано

Изм.	Лист	№ докум.	Подпись	Дата

последовательностью латинскими буквами, цифрами и знаком ‘_’ и должно начинаться со знаком ‘\$’

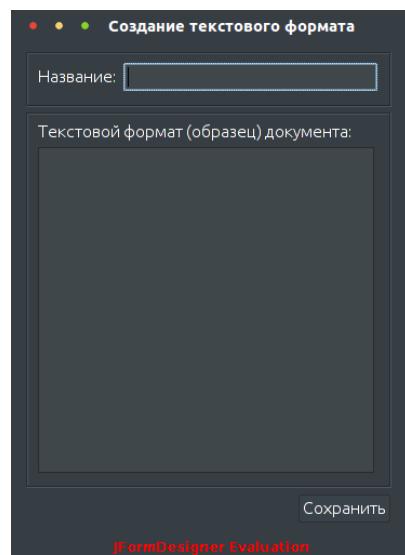


Рисунок 5.8 - Создание шаблона текстового формата

Чтобы сохранить шаблон, нажать на кнопку «Создать».

5.5. Запуск

Чтобы запустить работу программы, необходимо выбрать в **меню → распознавание → запустить.**

5.6. Просмотр промежуточных результатов

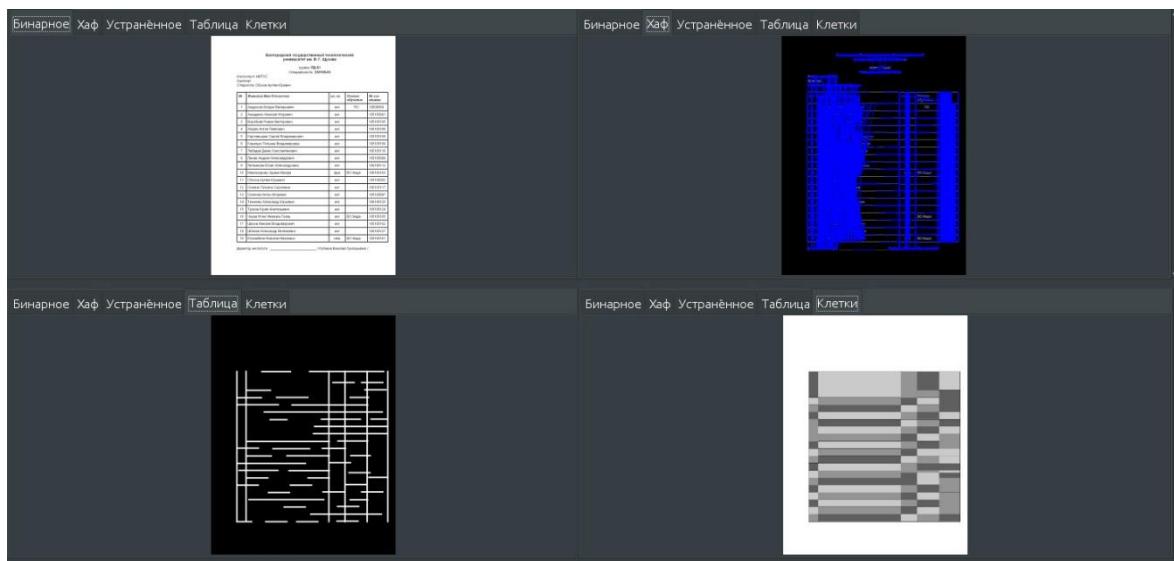


Рисунок 5.9 - Промежуточные результаты работы программы

Когда программа запускается, процесс решения задачи отражается в баре прогресса, а также последовательными промежуточными рисунками. Эти

Изм.	Лист	№ докум.	Подпись	Дата

рисунки, как на рисунке 5.9, показываются в каждом окне раздела промежуточных результатов:

- **Окно «бинарное»:** в этом окне показывается результат приведения в чёрно-белое, а затем в бинарное изображение.
- **Окно «Хаф»:** в этом окне показывается результат применения метода Кэнни и преобразования Хафа на бинарное изображение.
- **Окно «таблица»:** в этом окне показывается окончательный результат применения нескольких этапов перед тем чтобы вывести возможные составные ребра таблицы.
- **Окно «клетки»:** в этом окне показывается все обнаруженные клетки. Они накрашены разными цветами для наглядности.

Изм.	Лист	№ докум.	Подпись	Дата

6. Оценка экономической эффективности проекта

6.1. Назначение программного продукта

Несмотря на то что текущее общество считается очень развитым, нельзя отрицать необходимости в бумажных носителях. Банки ещё выдают выписку как подтверждение об осуществленных транзакциях компании или личных лиц. Во многих процедурах оформления документов, зачастую требуется скан паспорта. В приёмной комиссии разных университетов требуют справки об успеваемости, копии диплома, и т.д. Бумаги в наше время используются в качестве носителя информации, т.е. информацию в них записывают из какой-нибудь системы, и дальше либо проверяются, либо вводятся в другую систему. Ввод этих данных в настоящее время пока делается вручную. Вручной ввод данных занимает достаточно много времени. Текущий программный продукт создан для освобождения от вручного ввода данных или/и ускорения процесса ввод данных в вычислительную систему, либо просто их оцифровки для сохранения.

6.2. Расчет единовременных затрат на разработку ПО

К единовременным затратам разработчика относятся затраты на теоретические исследования, постановку задачи, проектирование, разработку алгоритмов и программ, отладку, опытные запуски, оформление документов, исследование рынка и рекламу.

Трудоемкость по стадиям проектирования представлена в виде таблицы 6.1

Общая трудоемкость разработки ПО рассчитывается по формуле:

$$T_{об} = \sum_{i=1}^n T_i$$

где $T_{об}$ — общая трудоемкость разработки, дн;

T_i — трудоемкость по стадиям, дней;

n — количество стадий разработки.

Изм.	Лист	№ докум.	Подпись	Дат	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА		
					Лит.	Лист	Листов
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	76	116
Руковод.	Брусенцев А.И.						
Консул.	Гриненко Г.П.						
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.						
БГТУ им. В. Г. Шухова, ПВ-51							

$$T_{об} = 5 + 10 + 15 + 40 + 12 = 82$$

Таблица 6.1 - Содержание стадий научно-исследовательской работы (НИР)

Стадия НИР	Содержание работ	Трудоемкость	
		Дни	%
Техническое задание	Подбор и изучение литературы, анализ составления вопроса, согласование с руководителем и утверждение технического задания и плана работ. Обоснование принципиальной возможности решения поставленной задачи. Постановка задачи.	5	6
Эскизный проект	Теоретическая разработка темы. Предварительная разработка структуры входных и выходных данных. Разработка общего описания алгоритма решения задачи.	10	12
Технический проект	Проектирование. Определение основных блоков, классов, объектов. Разработка сценария взаимодействия объектов.	15	18
Рабочий Проект	Написание и отладка программ. Тестирование и сборка системы.	40	49
Внедрение	Подготовка инструкций пользователям, написание, оформление и защита отчета (дипломного проекта). Регистрация.	12	15
Итого:		Toб = 82	100

В смету затрат на разработку ПО включаются:

- материальные затраты;
- основная и дополнительная зарплата;
- отчисления на социальные нужды;
- стоимость машинного времени на подготовку и отладку программ;
- стоимость инструментальных средств;
- накладные расходы.

Далее представлен расчет сметы затрат на разработку ПО.

6.2.1. Материальные затраты

Таблица 6.2 - Материальные затраты

Бумага SvetоСopy 500 листов	1	150	150
Картридж для принтера HP LaserJet 1018	1	1500	1500
Материальные затраты			1,650.00

К материальным затратам относятся стоимость всех материалов, которые будут использованы в процессе разработки ПО в действующих ценах, представим данные в виде таблицы 6.2:

6.2.2. Основная и дополнительная заработная плата

Основная заработная плата $Z_{осн}$ включает заработную плату всех, кто принимает участие в разработке ПО. В данном случае это разработчик, руководитель проекта и консультант по экономической части. Основная заработная плата каждого участника разработки будет рассчитываться по формуле:

$$Z_{осн} = T_{об} * Z_{ср.дн.},$$

где $Z_{ср.дн.}$ - среднедневная зарплата одного работника, денежные единицы, руб.; $T_{об}$ - общая трудоемкость проекта, дни.

Для разработчика ПО трудоемкость принимаем по данным таблицы 6.1 (т.е. 82 дней), для руководителя — 23 часов, для консультанта по экономической части — 2 час, для консультанта по программированию — 3 час, для консультанта по нейронным сетям — 4 час.

Дополнительная зарплата рассчитывается, как 10% к основной. Заработная плата составляет 150 руб./ч. В итоге основная заработная плата при разработке отображена в таблице 6.3:

Таблица 6.3 - Зарплаты

	ч	150.00 ₽/час
Руководитель	23	3,450.00 ₽
консультант экономики	2	300.00 ₽
Консультант по программированию	3	450.00 ₽
консультант нейронные сети	4	600.00 ₽
Разработчик	5ч/дн x 82дн	61,500.00 ₽
Итог		66,300.00 ₽

Дополнительная заработная плата равна 10% от основной зарплаты:

$$Z_{доп} = 66300 * 0,1 = 6630 \text{ руб}$$

Итого, общая заработная плата составляет:

$$Z_{общ} = Z_{осн} + Z_{доп} = 66300 + 6630 = 72930 \text{ руб}$$

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						78

6.2.3. Отчисления на социальные нужды

Отчисления на социальные нужды принимаются по законодательству на момент выполнения дипломного проекта (на 2014г.: 30% от общего фонда заработной платы):

$$\text{Соц} = Z_{\text{общ}} * 30\% = 18961 \text{ руб}$$

6.2.4. Стоимость машинного времени

Стоимость машинного времени зависит от себестоимости машино-часа работы ЭВМ, а также времени работы ЭВМ, и включает амортизацию ЭВМ и оборудования, затраты на электроэнергию.

$$A_m = \frac{O_\phi \cdot H_{am}}{365 \cdot 100} \cdot T_m$$

где A_m — амортизационные отчисления, руб.;

O_ϕ — стоимость ЭВМ и оборудования, руб.;

H_{am} — норма амортизации, %;

T_m — время использования оборудования, дни, равное:

$$T_m = 0,35 * T_{\text{экс}} + 0,6 * T_{\text{тех. пр}} + 0,8 * T_{\text{раб. пр}} + 0,6 * T_{\text{вн}},$$

$T_{\text{экс}}$, $T_{\text{тех. пр}}$, $T_{\text{раб. пр}}$, $T_{\text{вн}}$ — фактические затраты времени на разработку эскизного, технического, рабочего проекта и внедрение в днях.

Далее рассчитывается общее время разработки:

$$T_m = 0.35 * 10 + 0.6 * 15 + 0.8 * 40 + 0.6 * 12 = 51.7 \text{ ч}$$

Список использованного оборудования, цена, количество и конечная стоимость представлены в таблице 6.4.

Амортизация всего оборудования за год:

$$A_m = \frac{25000 * 30 * 51.7}{250 * 100} = 1551 \text{ руб}$$

Таблица 6.4 - Список используемого оборудования

Наименование оборудования	Цена, руб.	Количество, шт.	Стоимость, руб.
Персональный компьютер	25000	1	25000

Затраты на электроэнергию для всего оборудования:

$$Z_{эл} = C_{эл} * M_{ЭВМ} * T_M * T_{сут},$$

где $C_{эл}$ – стоимость 1 кВт/ч электроэнергии, (3,25 руб.); $M_{ЭВМ}$ – мощность ЭВМ, кВт/ч, (всего 0,4 кВт/ч); $T_{сут}$ – суточное время работы ЭВМ в часах.

$$Z_{эл} = 3,25 * 0,4 * 51,7 * 5 = 336 \text{ руб}$$

Следовательно, затраты на оплату машинного времени составляют:

$$Z_M = A_M + Z_{эл} = 1551 + 336 = 1887 \text{ руб}$$

6.2.5. Стоимость инструментальных средств

Стоимость инструментальных средств включает в себя стоимость системного ПО, используемого при разработке проекта в размере износа за период. Стоимость инструментальных средств представлена в таблице 6.5.

Лицензия была рассчитана за период использования ПО (4,1 мес.)

Таблица 6.5 - Список используемых инструментальных средств

JFormDesigner	4,1 мес.	8000 руб./год	2,733.33 ₽
IntelliJ	4,1 мес.	10000 руб./год	3,416.67 ₽
Microsoft Office	4,1 мес.	16000 руб./год	5,466.67 ₽
Java 7		Бесплатно	0 ₽
Tesseract		Бесплатно	0 ₽
Hibernate		Бесплатно	0 ₽
OpenCV		Бесплатно	0 ₽
Стоимость инструментальных средств			11,616.67 ₽

Амортизация инструментальных средств составит:

$$A_i = \frac{C_i * 30 * T_M}{250 * 100} = \frac{11616 * 30 * 51.7}{250 * 100} = 720 \text{ руб}$$

6.2.6. Затраты на интернет

Во время разработки, стоимость подключения к интернету стоит 400руб/мес. При это, затраты на интернет: $Z_{инт} = 400 * 4,1 = 1640 \text{ руб}$

6.2.7. Накладные расходы

Накладные расходы определяются в размере 20-60% от основной заработной платы разработчиков ПС. Они принимаются равными 20%:

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						80

$$C_{\text{нр}} = Z_{\text{очн}} * 0,2 = 66300 * 0,2 = 13262 \text{ руб}$$

6.2.8. Смета затрат на разработку ПО

На основании проведенных расчетов в пункте 2 составим смету затрат на разработку ПО

$$C_{\text{маш}} = \frac{A_m + Z_{\text{эл}} + Ц_i}{T_m} = \frac{1151 + 336 + 12337}{51,7} = 275$$

, где $Ц_i$ — затраты на инструментальные средства.

Себестоимость ПО — $C_{\text{по}} = 120394,52$ руб.

Прибыль — рассчитывают по формуле:

$$\Pi = C_{\text{по}} * 0,3 = 120394,52 * 0,3 = 36118,36 \text{ руб.}$$

Налог на добавленную стоимость (20%) :

$$\text{НДС} = (C_{\text{по}} + \Pi) * 0,2 = (120394,52 + 36118,36) * 0,2 = 31302,57 \text{ руб.}$$

Цена программного обеспечения

$$Ц_{\text{по}} = C_{\text{по}} + \Pi + \text{НДС} = 121115 + 36335 + 31490 = 187,815.45 \text{ руб.}$$

Таблица 6.6 - Смета затрат на разработку ПО

Элементы затрат	Стоимость
Материальные затраты	1,650.00
Основная и дополнительная зарплата	72,930.00
Отчисления на социальные нужды	18,961.80
Оплата машинного времени	336.05
Стоимость инструментальных средств	11616.16
Затраты на Интернет ресурсы	1,640.00
Накладные расходы	3,260.00
Всего	120394.52

6.3. Инвестиционный план

Далее будет составлен инвестиционный план. Распределяется итоговая сумма затрат на разработку по этапам проектирования пропорционально трудоемкости (см. таблицу 6.7).

Финансирование разработки программного продукта будет осуществляться за счет собственных средств без использования заемных. У разработчика в наличии уже имеются перечисленная компьютерная техника и

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						81

инструментальные средства. Оплату труда руководителя и консультанта по экономической части осуществляет университет.

Таблица 6.7 - План инвестиций

Этапы проектирования	Месяцы				
	1	2	3	4	5
Техническое задание	7,385				
Эскизный проект	14,770				
Технический проект	7,385	14,770			
Рабочий проект		14,770	29,540	14,770	
Внедрение				14,770	2,954
Итого:	29,540	29,540	29,540	29,540	2,954

6.4. График реализации проекта.

Таблица 6.8 - График реализации проекта

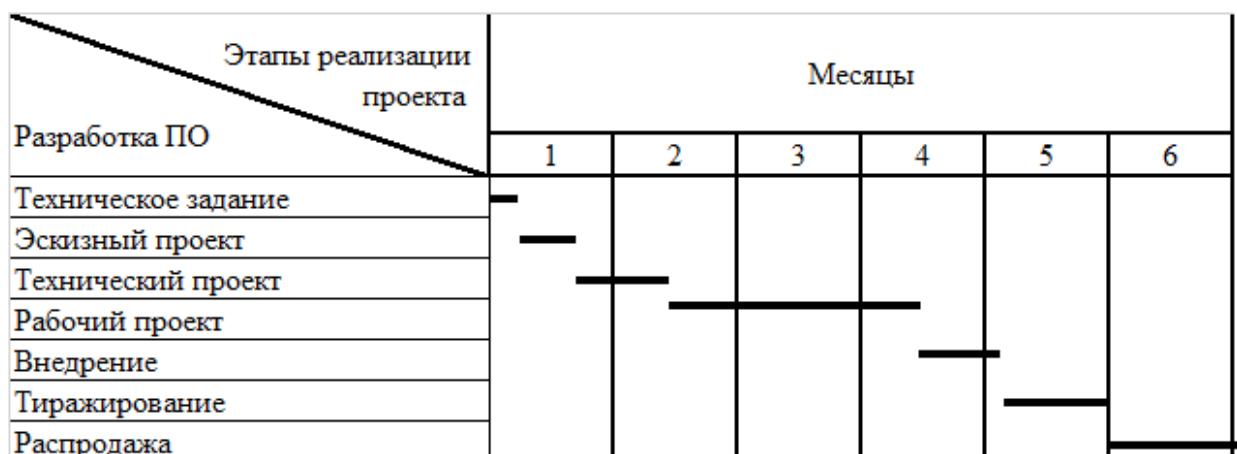


Таблица 6.9 – План возврата кредита

№ платежа	Дата платежа	Сумма платежа	Основной долг	Начисленные проценты	ком.	Остаток
1	Июнь, 2015	5 415,50	4 665,50	750,00	0,00	55 334,50
2	Июль, 2015	5 415,50	4 723,82	691,68	0,00	50 610,68
3	Август, 2015	5 415,50	4 782,87	632,63	0,00	45 827,82
4	Сентябрь, 2015	5 415,50	4 842,65	572,85	0,00	40 985,17
5	Октябрь, 2015	5 415,50	4 903,18	512,31	0,00	36 081,98
6	Ноябрь, 2015	5 415,50	4 964,47	451,02	0,00	31 117,51
7	Декабрь, 2015	5 415,50	5 026,53	388,97	0,00	26 090,98
8	Январь, 2016	5 415,50	5 089,36	326,14	0,00	21 001,62
9	Февраль, 2016	5 415,50	5 152,98	262,52	0,00	15 848,64
10	Март, 2016	5 415,50	5 217,39	198,11	0,00	10 631,25
11	Апрель, 2016	5 415,50	5 282,61	132,89	0,00	5 348,64
12	Май, 2016	5 415,50	5 348,64	66,86	0,00	0,00
Итого по кредиту		64 985,98	60 000,00	4 985,98	0,00	

При составлении графика реализации проекта необходимо учесть время на разработку ПО по стадиям: техническое задание, эскизный проект, технический проект, рабочий проект, внедрение; а также время на покупку оборудования и инструментальных средств, обучение персонала и выход на эксплуатацию.

Для инвестиции проекта взять кредит 60000руб, и собственная сумма 60,394.52руб, под процентом 15%

6.5. Анализ рыночных возможностей программного продукта

Разрабатываемый продукт предназначен для компаний, которые работают с большим количеством данных, сохраненных на физическом носителе или полученных из документов в бумажном виде. Таким образом, главными покупателями данного продукта будут являться корпорации.

Географически рынок представляет собой весь мир.

Для сегментации рынка можно выбрать в качестве показателя количество обрабатываемых данных в приёмном отделении компании. Очевидно, что необходимость в таком приложении для организации различна, поэтому для каждого сегмента можно использовать свою цену, при этом создавая вариативные наборы функций для каждого из них. Целесообразно разделить рынок на два сегмента: крупные организации и остальные организации. К крупными относятся банки, университеты, мобильные операторы, почты, гипермаркеты и т.д. К маленьким относятся школы, рестораны, центр развлечения и т.д.

Среди ИТ-компаний, производящих приложения для распознавания текстов, ABBYY является единственной, которая производит приложение, умеющее работать со структурированными документами.

ABBYY FineReader переводит изображения документов и любые типы PDF-файлов в электронные редактируемые форматы, определяет и точно восстанавливает логическую структуру документа в его электронной копии.

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						83

Бессрочная лицензия полной версии программы стоит 5990руб, а годовая - 2390руб/год.

ABBYY Business Card Reader автоматически определяет имя, фамилию, номер телефона, адрес и другую информацию и сохраняет ее в соответствующие поля базы данных. Программа стоит 1700 руб.

Кроме ABBYY мало таких компаний, поэтому конкурентов на этом рынке мало.

6.6. План тиражирования и реализации программного продукта

Далее следует план по тиражированию и реализации ПО всему миру

Таблица 6.10 - План по реализации ПО

Показатели	месяц							Год	
	6	7	8	9	10	11	12	2	3
Объем Тиражирования, шт.	10	15	25	35	40	40	45	500	650
Цена за использование ПО на год для 1-го комп, руб.	17600	22000	2200	2200	2200	2200	2200	2200	2200
Выручка от налогов, руб.	26400	33000	2200	44000	55000	2200	61600	77000	2200
выручка от реализации									
Объем Тиражирования, шт.	5	10	10	20	20	25	30	360	400
Цена за использование ПО на год для 1-го комп, руб.	12000	15000	1500	1500	1500	1500	1500	1500	1500
Выручка от налогов, руб.	12000	15000	1500	1500	1500	1500	1500	1500	1500
выручка от реализации									
Итого: выручка от реализации	38400	56000	85600	94400	100400	115200	1312000	1624000	
	23600	6000	7500	1500	30000	36000	432000	480000	

СЕГМЕНТ 1

СЕГМЕНТ 2

6.7. Смета затрат на тиражирование и рекламу

Затраты на тиражирование представлены в таблице 6.11.

Таблица 6.11 - Смета затрат на тиражирование

Показатели	месяц								год	
	6	7	8	9	10	11	12			
Затраты на тиражирование	2500	1000	1500	2500	1000	1500	2500	1000	1500	2500
Затраты на рекламу										
Итого:	2500	1000	1500	2500	1000	1500	2500	1000	1500	2500

6.8. План прибыли от продаж

План прибыли от продаж представлен в таблице 6.12.

Таблица 6.12 - План прибыли от продаж

Показатели	месяц								год	
	6	7	8	9	10	11	12			
Выручка от реализации и сопровождения	20880	5220	26100	2500	23600	23600	23600	23600	23600	23600
Затраты на тиражирование и рекламу	32720	8180	40900	2500	38400	38400	38400	38400	38400	38400
Прибыль валовая	46800	11700	58500	2500	56000	56000	56000	56000	56000	56000
Налог (20%)	70480	17620	88100	2500	85600	85600	85600	85600	85600	85600
Прибыль чистая	82320	20580	102900	2500	94400	94400	94400	94400	94400	94400

6.9. Финансовый план проекта

Для оценки финансовой состоятельности проекта далее будут представлены исходные и расчётные данные в Error! Reference source not found.

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						85

Таблица 6.13 - Оценка финансовой состоятельности проекта, руб.

показатели	период												
	год												
	1												2
	1	2	3	4	5	6	7	8	9	10	11	12	3
Затраты на разработку ПО	I. Инвестиционная деятельность												
техническое задание,	-29540			7,385	14,770	7,385							
эскизный проект,	-29540			14,770	14,770								
технический проект,	-29540			29,540									
рабочий проект	-29540			14,770	14,770								
внедрение	-2954			2,954									
Итого: эффект от инвестиций. Деятельности	0	0	0	0	0	0	0	0	0	0	0	0	0
II. Операционная деятельность													
показатели	период												
	год												
	1												2
	1	2	3	4	5	6	7	8	9	10	11	12	3
Выручка, всего,	0	0	0	0	0	0	0	0	0	0	0	0	0
Затраты, всего, в том числе:	26100	-258,5	2500	23600									
амортизация оборудования	40900	-258,5	2500	38400									
Прибыль валовая	58500	-258,5	2500	56000									
	88100	-258,5	2500	85600									
	96900	-258,5	2500	94400									
	102900	-258,5	2500	100400									
	117700	-258,5	2500	115200									
	1342000	-1551	30000	1312000									
	1654000	-1551	30000	1624000									

*Таблица 6.13 - Оценка финансовой состоятельности проект
(продолжение)*

Налог на прибыль		0	0	29540	0	0	0
Прибыль чистая		0	0	29540	0	0	0
Итого: эффект от операционной деятельности		61314	0	60000	1314	0	0
III. Финансовая деятельность							
Собственные средства		-29540	0	0	0	0	0
Кредит		-2954	0	0	0	0	0
Возврат кредита		15206	-5415.5	-5,415.50	0	0	20880
Итого: эффект от финансовой деятельности		2234	-720	27046	-5415.5	-5,415.50	5220
IV. Сальдо денежной наличности (I+II+III)							
82658		41126	-5415.5	-5,415.50	0	0	32461.5
147464		64806	-5415.5	-5,415.50	0	0	46541.5
219310		71846	-5415.5	-5,415.50	0	0	70221.5
295956		76646	-5415.5	-5,415.50	0	0	77261.5
384442		88486	-5415.5	-5,415.50	0	0	70480
1418582.5		1034140.5	-37908.5	-37,908.50	0	0	17620
2740231.5		1321649	0	0	0	0	1323200
IV. Сальдо денежной наличности (I+II+III)							
ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА							

6.10. Показатели экономической эффективности проекта

Международная практика в процессе оценки проектов использует несколько обобщающих показателей. К таким показателям относятся:

- интегральный экономический эффект (чистая текущая стоимость);
- индекс доходности;
- внутренний коэффициент эффективности;

- максимальный денежный отток;
 - период возврата капитальных вложений и срок окупаемости.

6.10.1. Интегральный экономический эффект

Определяется NPV, чистая текущая стоимость проекта, путем вычисления совокупного дохода за весь период функционирования проекта и всех видов расходов, суммированных за тот же период с учетом дисконтирования. Расчет NPV представлен в таблице 6.13. Чистый денежный поток (ЧДП) рассчитывается как разница между результатами операционной и инвестиционной деятельности.

Таблица 6.13 - Денежные потоки, руб.

Коэффициент дисконтирования определяется по формуле:

$$\alpha_t = \frac{1}{(1+r)^t}$$

<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА				

где r – ставка дисконтирования.

Ставка дисконтирования рассчитывается как:

$$r_{год} = \frac{\text{ставка рефинансирования} + 1}{\text{инфляция в год} + 1} - 1 + \text{риск}$$

При ставке рефинансирования = 10%, инфляции=13%, риске=15% получим $r_{год}= 0,1235$.

Дисконтированный денежный поток (ДДП) рассчитывается как произведение чистого денежного потока на коэффициент дисконтирования по каждому периоду.

Расчет интегрального экономического эффекта NPV можно представить в виде формулы:

$$NPV = \sum_{t=1}^n (\Delta_t - Z_t + A_t) \cdot \alpha_t - \sum_{t=1}^n K_t \cdot \alpha_t \sum_{t=1}^n (\Pi_{Ч_t} + A_t) \cdot \alpha_t - \sum_{t=1}^n K_t \cdot \alpha_t$$

где: Δ_t – доходы от реализации проекта по годам; Z_t – текущие затраты без амортизации; A_t – амортизационные отчисления; K_t – капитальные вложения в основные и оборотные фонды; $\Pi_{Ч}$ – прибыль чистая.

$NPV = 3095563.407$ руб. > 0 , проект эффективен.

6.10.2. Индекс доходности (SRR)

Определяется как отношение суммарного дисконтированного дохода к суммарным дисконтированным капитальным вложениям:

$$SRR = \frac{\sum_{t=1}^n (\Pi_{Ч_t} + A_t) \cdot \alpha_t}{\sum_{t=1}^n K_t \cdot \alpha_t}$$

Таким образом, $SRR=3.25>1$, то можно сделать вывод об эффективности проекта.

6.10.3. Внутренний коэффициент эффективности проекта (IRR)

Определяется как пороговое значение рентабельности, при котором NPV равно нулю.

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						89

$$r_{\text{нор}} = r_1 + \left[\frac{NPV_{r1}}{(NPV_{r1} - NPV_{r2})} \right] \cdot (r_2 - r_1),$$

где r_1 – исходная ставка дисконтирования; r_2 – ставка дисконтирования, при которой $NPV > 0$; $r_{\text{нор}}$ – внутренний коэффициент эффективности проекта; NPV_{r1} , NPV_{r2} – NPV соответственно при r_1 и r_2 .

Для расчета IRR ($r_{\text{нор}}$) выбирают два значения коэффициента дисконтирования $r_1 < r_2$ таким образом, чтобы на интервале r_1-r_2 , NPV меняла знак с положительного на отрицательный и наоборот.

При $r_1 = 0.99009901$, $NPV_1 = -29247.52$ руб.

При $r_2 = 0.91433982$, $NPV_2 = 38815.72$ руб.

Проект считается эффективным если $r_{\text{нор}} > r_1$.

$$r_{\text{нор}} = 0.99009901 + \left[-\frac{29247.52}{-29247.52 - 38815.72} \right] * (0.99009901 - 0.905286955) = 1.265436$$

Проект эффективен.

Срок возврата капитальных вложений и срок окупаемости могут быть определены аналитическим и графическим способами:

а) аналитический способ:

$$T_{\text{воз}} = t_x + \frac{|NPV_t|}{ДДП_{t+1}},$$

где t_x – количество периодов, при которых $NPV < 0$; NPV_t – величина NPV в t -м периоде; $ДДП_{t+1}$ – величина $ДДП$ в “ $t+1$ ”-м периоде.

$$T_{\text{воз}} = 8 + \frac{|-25390.58469|}{38815.72928} = 8,65 \text{ месяца}$$

Таким образом, $T_{\text{воз}} \approx 8$ месяцев и 18 дней.

Период окупаемости проекта:

$$T_{\text{возв}} = T_{\text{ок}} - T_{\text{ин}},$$

Таким образом, $T_{\text{ок}} = 8,65$ месяцев – 5 месяца = 3 месяца и 18 дней.

Изм.	Лист	№ докум.	Подпись	Дата

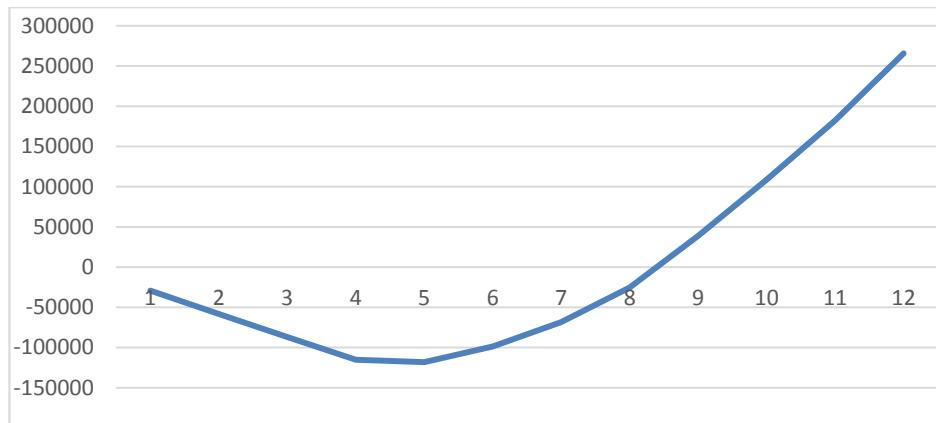


Рисунок 6.1 - Финансовый профиль проекта

б) графический способ на основе построения финансового профиля проекта; финансовый профиль проекта представляет собой график изображения величины кумулятивной чистой текущей стоимости во времени (см. рисунок 6.1).

6.11. Правовые аспекты

1.1. Настоящее Лицензионное соглашение (далее — Лицензия) устанавливает условия использования программы «Struct OCR» (далее — Программа) и заключено между любым лицом, использующим Программу (далее — Пользователь) и Манитраиву Адама Маэфа, являющимся правообладателем исключительных прав на Программу (далее — Правообладатель).

1.2. Копируя Программу, устанавливая её на свой компьютер или используя Программу любым образом, Пользователь выражает свое полное и безоговорочное согласие со всеми условиями Лицензии.

1.3. Использование Программы разрешается только на условиях настоящей Лицензии. Если Пользователь не принимает условия Лицензии в полном объёме, Пользователь не имеет права использовать Программу в каких-либо целях. Использование Программы с нарушением (невыполнением) какого-либо из условий Лицензии запрещено.

1.4. Использование Программы Пользователем на условиях настоящей Лицензии в личных некоммерческих целях осуществляется безвозмездно. Использование Программы на условиях и способами, не предусмотренными

Иzm.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						91

настоящей Лицензией, возможно только на основании отдельного соглашения с Правообладателем.

6.11.1. Лицензионное соглашение

2.1. Правообладатель безвозмездно, на условиях простой (неисключительной) лицензии, предоставляет Пользователю непередаваемое право использования Программы на территории всех стран мира следующими способами:

2.1.1. Применять Программу по прямому функциональному назначению, в целях чего произвести её копирование и установку (воспроизведение) на компьютер(ы) Пользователя. Пользователь вправе произвести установку Программы на три компьютера.

2.1.2. Воспроизводить и распространять Программу в некоммерческих целях (безвозмездно).

2.2. Программа должна использоваться (в том числе распространяться) под наименованием: «Struct OCR». Пользователь не вправе изменять и/или удалять наименование Программы, знак охраны авторского права (copyright notice) или иные указания на Правообладателя.

6.11.2. Авторское право

3.1. Исключительное право на Программу принадлежит Правообладателю.

6.11.3. Ограничения

4.1. За исключением использования в объемах и способами, прямо предусмотренными настоящей Лицензией или законодательством РФ, Пользователь не имеет права изменять, декомпилировать, дизассемблировать, дешифровать и производить иные действия с объектным кодом и исходным текстом Программы, имеющие целью получение информации о реализации алгоритмов, используемых в Программе, создавать производные произведения с использованием Программы, а также осуществлять (разрешать осуществлять) иное использование Программы, любых компонентов

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						92

Программы, хранимых Программой на мобильном устройстве Пользователя картографических материалов, иных изображений и прочих данных, без письменного согласия Правообладателя.

4.2. Пользователь не имеет право воспроизводить и распространять Программу в коммерческих целях (в том числе за плату), в том числе в составе сборников программных продуктов, без письменного согласия Правообладателя.

4.3 Пользователь не имеет права распространять Программу в виде, отличном от того, в котором он её получил, без письменного согласия Правообладателя.

6.11.4. Ответственность Производителя

5.1. Программа предоставляется на условиях "как есть" (as is). Правообладатель не предоставляет никаких гарантий в отношении безошибочной и бесперебойной работы Программы или отдельных её компонентов и/или функций, соответствия Программы конкретным целям Пользователя, не гарантирует достоверность, точность, полноту и своевременность Данных, а также не предоставляет никаких иных гарантий, прямо не указанных в настоящей Лицензии.

5.2. Правообладатель не несет ответственности за какие-либо прямые или косвенные последствия какого-либо использования или невозможности использования Программы (включая Данные) и/или убытки, причиненные Пользователю и/или третьим сторонам в результате какого-либо использования, неиспользования или невозможности использования Программы (включая Данные) или отдельных её компонентов и/или функций, в том числе из-за возможных ошибок или сбоев в их работе, за исключением случаев, прямо предусмотренных законодательством.

Изм.	Лист	№ докум.	Подпись	Дата	ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА	Лист
						93

Заключение

В ходе выполнения дипломного проекта было реализовано программное обеспечение для оцифровки табличных структур с бумажных носителей на основе алгоритмов искусственных нейронных сетей. Были выполнены следующие задачи:

- изучены алгоритм метода Кэнни и преобразования Хафа;
- изучена библиотека OpenCV;
- изучены алгоритм нейронной сети, библиотека Tesseract и обучение её внутренней нейронной сети;
- разработаны алгоритмы преобразования изображения в чёрно-белое и в бинарное;
- разработаны алгоритмы кластеризации K-Means
- разработаны алгоритм нечёткого поиска строки с помощью оценки по расстоянию Левенштейна.
- разработаны алгоритмы для обнаружения наклона документа, нахождения составных ребер таблицы и нахождения всех клеток таблицы;
- проведена отладка и тестирование проекта. Все выявленные недочеты были устранены.
- оценена экономическая эффективность проекта. Было выявлено, что данный продукт будет выгоден в анализированном рынке. К тому же, средства, потраченные на разработку, быстро окупаются.

Таким образом, можно сделать вывод, что цель дипломного проекта была выполнена, реализованный проект является актуальным, рентабельным, полезным.

Изм.	Лист	№ докум.	Подпись	Дат	ЗАКЛЮЧЕНИЕ			
Разраб.	Манитрапиву А.М.				Программная реализация		Лит.	Лист
Руковод.	Брусенцев А.И.				алгоритмов на основе		Листов	
Консул.					искусственных нейронных сетей			
Н. контр.	Полунин А.И.				для оцифровки табличных		94	116
Зав. каф.	Поляков В.М.				структур с бумажных носителей			
							БГТУ им. В. Г. Шухова, ПВ-51	

Список литературы

1. Пожар в библиотеке ИНИОН РАН [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=71343477> (дата обращения: 01.03.2015).
2. Институт научной информации по общественным наукам (ИНИОН РАН) [электронный ресурс] // Официальный сайт ИНИОН РАН. Режим доступа: <http://www.inion.ru/> (дата обращения: 01/03/2015).
3. Всемирный фонд дикой природы [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=71100927> (дата обращения: 29.05.2015).
4. Цветовая модель [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=69710654> (дата обращения: 01.04.2015).
5. Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов [текст] / Р. Блейхут. – М.: // Мир, 1998.
6. Грузман И.С., Киричук В.С., Косых В.П., Перетягин Г.И., Спектор А.А. Цифровая обработка изображения в информационных системах [текст] // Новосибирский государственный технический университет 2000 г. 352 с.
7. В.Т. Фисенко, Т.Ю. Фисенко Компьютерная обработка и распознавание изображений [текст] // СПб: СПбГУ ИТМО, 2008. – 192 с.
8. Детектор границ Кэнни. [Электронный ресурс] // habrahabr – Режим доступа: <http://habrahabr.ru/post/114589/> (дата обращения: 15.04.2013)
9. Оператор Кэнни [электронный ресурс] // Википедия. [2015—2015].. Режим доступа: <http://ru.wikipedia.org/?oldid=67841167> (дата обращения: 15.04.2015).
10. Преобразование Хафа [электронный ресурс] // Википедия. [2015—2015]. Режим доступа: <http://ru.wikipedia.org/?oldid=69986255> (дата обращения: 14.04.2015).

Изм.	Лист	№ докум.	Подпись	Дат	СПИСОК ЛИТЕРАТУРЫ		
					Лит.	Лист	Листов
Разраб.	Манитрапиву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	95	116
Руковод.	Брусенцев А.И.						
Консул.							
Н. контр.	Полунин А.И.						
Зав. каф.	Поляков В.М.						
БГТУ им. В. Г. Шухова, ПВ-51							

11. Заде Л.А. Размытые множества и их применение в распознавании образов и кластер-анализе [текст] // Классификация и кластер / Пер. с англ. ; Под ред. Дж. Вэн Райзина. - М., 1980. - с. 208-247.
12. Градиентный спуск [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=70087855> (дата обращения: 19.04.2015)
13. Donald Olding Hebb. The Organization of Behavior: A Neuropsychological Theory. [текст] // Wiley, 1949. — 335 с.
14. Spacial filters – Convolution [электронный ресурс] // Image Processing Learning Ressource. Режим доступа: <http://homepages.inf.ed.ac.uk/rbf/HIPR2/convolve.htm> (дата обращения: 01.04.2015).
15. Оператор Кэнни [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=67841167> (дата обращения: 01.04.2015).
16. Hough Transform [электронный ресурс] // OpenCV 2.4. documentation. Режим доступа: http://docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html (дата обращения: 01.04.2015).
17. В.Д. Дмитриенко, Н.И. Корсунов. Основы теории нейронных сетей [текст] // Белгород: БИИММАП, 2001 – 159с.
18. Расстояние Левенштейна [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=69269981> (дата обращения: 10.05.2015).
19. OpenCV [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=71235048> (дата обращения: 15.05.2015).
20. Feature detection [электронный ресурс] // OpenCV 2.4. documentation. Режим доступа: http://docs.opencv.org/modules/imgproc/doc/feature_detection.html (дата обращения: 01.04.2015).

Изм.	Лист	№ докум.	Подпись	Дата

СПИСОК ЛИТЕРАТУРЫ

- 21.Tesseract [электронный ресурс] // Википедия. Режим доступа:
<http://ru.wikipedia.org/?oldid=70781648> (дата обращения: 13.05.2015).
- 22.Tess4j – JNA Wrapper for Tesseract [электронный ресурс] // Tess4J API Documentation. Режим доступа: <http://tess4j.sourceforge.net/docs/docs-2.0/> (дата обращения: 13.05.2013)
- 23.Hibernate (библиотека) [электронный ресурс] // Википедия. Режим доступа: <http://ru.wikipedia.org/?oldid=71187081> (дата обращения: 01.05.2015).

Изм.	Лист	№ докум.	Подпись	Дата

СПИСОК ЛИТЕРАТУРЫ

Лист
97

Приложение А

Техническое задание в соответствии с ГОСТ

Основания для разработки

Основанием для разработки является дипломный проект по специальности 230105.65 «Программное обеспечение вычислительной техники и автоматизированных систем».

Тема проекта согласуется с кафедрой и утверждается приказом ректора. Организация, утверждающая документ: Белгородский Государственный технологический университет им. В. Г. Шухова. Адрес: РФ, г. Белгород, ул. Костюкова, д. 46.

Назначение разработки

Функциональное назначение программного продукта

Приложение предназначено для распознавания и автоматической обработки данных из бумажных носителей (квитанция, выписка банковского счёта или транзакции, паспорт, чек и т.д.) и введения их в базу данных или преобразования в CSV, SQL или XML-файл, соответствующий распознанной таблице из входного документа, целью которого является освобождение пользователей от ручного ввода данных.

Эксплуатационное назначение

Приложение должно применяться в отделах компании, где необходимо часто вручную вводить данные из бумажных носителей. Например, в отделе приёма клиентов и в бухгалтерском отделе.

Также программа должна широко применяться на предприятиях, где нужно оцифровать бумажные архивы, в которых есть не только тексты, но также структурированные данные и транзакции в таблице.

Изм.	Лист	№ докум.	Подпись	Дат	ПРИЛОЖЕНИЕ А				
					Лит.	Лист	Листов		
Разраб.	Манитрапиев А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей				
Руковод.	Брусенцев А.И.					98	116		
Консул.					БГТУ им. В. Г. Шухова, ПВ-51				
Н. контр.	Полунин А.И.								
Зав. каф.	Поляков В.М.								

Возможными конечными пользователями являются также люди, которым нужна автоматизированная система для оцифровки документов с определёнными структурами.

Требования к программе или программному изделию

Требования к функциональным характеристикам

Приложение должно:

- Обеспечить возможность работать с изображениями, полученными от сканирования документов и находящимися в локальной системе или снятыми фотоаппаратом мобильного телефона;
- Обеспечить возможность работы одновременно с несколькими изображениями одинаковой структуры из указанной пользователем папки;
- Обеспечить возможность определения структуры таблицы и требуемого формата текстов в каждой ячейке таблицы;
- Иметь простой и интуитивный интерфейс и сообщать пользователю о развитии распознавания при работе алгоритма;
- Сохранять определённую структуру таблицы и снова загружать её из файла при последующей необходимости;
- Обеспечить возможность сохранения полученных данных от распознавания в CSV или XML-файл
- Обеспечить возможность автоматического ввода данных в базу данных, проверку выполненного ввода и отмену при необходимости;
- Обеспечить возможность создания MySql-скрипта, при запуске которого в системе с СУБД Mysql автоматически вводятся данные;
- Обеспечить повторное выполнение последней операции над другими изображениями;

Изм.	Лист	№ докум.	Подпись	Дата

Входными данными являются изображения, полученные от сканирования бумажного документа или от их фотографирования с помощью фотоаппарата мобильного телефона.

Выходными данными являются файлы следующего типа: CSV, SQL или XML-файл. Выходными данными также является ввод данных в базу данных.

Требования к надежности

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением совокупности организационно-технических мероприятий, перечень которых приведен ниже:

- минимальной процессорной мощностью компьютера или мобильного телефона, в котором приложение будет запускаться;
- минимальным количеством памяти, предназначенной для программы, чтобы можно было быстро и надежно работать с изображениями высокого разрешения;
- правильным заданием корректной структуры таблицы и формата текстов, соответствующих документу перед запуском распознавания;
- минимальным размером шрифта текста в документе не менее 10;

Условия эксплуатации

Особых требований к климатическим условиям не имеется.

Пятно в документе должно быть минимальным.

Минимальное количество персонала, требуемого для работы приложения, должно составлять не менее 2 штатных единиц - администратор БД и пользователь программы.

Администратор БД должен иметь образование в сфере ИТ и опыт работы с администрированием СУБД MySql. В его обязанности будет входить:

- Определение формата входных данных;
- При необходимости связывание программы с БД и обеспечение ввода данных в правильную таблицу.

Изм.	Лист	№ докум.	Подпись	Дата

Требований к квалификации у пользователя нет, но предпочтительно, чтобы он прошёл обучение по использованию программы, ознакомился с методами указания формата документов и структуры таблицы.

Требования к составу и параметрам технических средств

В состав технических средств должны входить:

- рабочий компьютер, предпочтительно многоядерный для обеспечения эффективного распараллеливания распознавания.

Требования к информационной и программной совместимости

Пользовательский интерфейс должен быть удобным, интуитивно понятным и содержать подсказки.

Исходные коды программы должны быть реализованы на языке Java, 7-ой версии. В качестве интегрированной среды разработки программы должна быть использована среда IntelliJ IDEA 14 Community Version. Взаимодействие с СУБД и создание базы данных реализуется на языке MySQL. Установленная версия Java на рабочем компьютере должна быть Java 7 или более поздняя.

Настройку программы должен выполнять администратор.

Для удобства администрирования приложение должно иметь администраторский режим, позволяющий детально настроить все возможные форматы входных документов и соответствующие связи с базой данных. В пользовательском режиме пользователь может просто загрузить готовые форматы, созданные администратором.

Требования к маркировке и упаковке

Особых требований к маркировке и упаковке не имеется.

Требования к транспортированию и хранению

Особых условий транспортировки и хранения продукт не требует. Срок хранения и эксплуатации – не ограничен.

Требования к программной документации

Состав программной документации должен включать в себя:

- техническое задание;

Изм.	Лист	№ докум.	Подпись	Дата

- спецификацию;
- текст программы;
- описание программы;
- программу и методики испытаний;
- пояснительную записку;
- описание применения;
- руководство программиста;
- руководство администратора;
- Стадии и этапы разработки
- Разработка должна быть проведена в три стадии:
 - разработка технического задания;
 - рабочее проектирование;
 - внедрение;

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания. Здесь должны быть выполнены перечисленные ниже работы:

- постановка задачи;
- определение и уточнение требований к техническим средствам;
- определение требований к программе;
- определение стадий, этапов и сроков разработки программы и документации на неё;
- выбор языков программирования;
- согласование и утверждение технического задания

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

- разработка проекта - должна быть выполнена работа по программированию и отладке программы;
- разработка программной документации;

Изм.	Лист	№ докум.	Подпись	Дата

- испытания по работоспособности проекта – должны быть выполнены следующие виды работ: разработка, согласование и утверждение программы и методики испытаний; проведение приемо-сдаточных испытаний; корректировка программы и программной документации по результатам испытаний.

На стадии внедрения должен быть выполнен этап разработки

- подготовка и передача проекта

Изм.	Лист	№ докум.	Подпись	Дата

ПРИЛОЖЕНИЕ А

Приложение Б

Метод doInBackground() класса TCWorker

```
// Created: 06/06/2015
package com.maheffa.TabulatedOCR;
import com.maheffa.TabulatedOCR.DBManager.*;
import com.maheffa.TabulatedOCR.GUI.*;
import com.maheffa.TabulatedOCR.ImageProcessing.*;
import com.maheffa.TabulatedOCR.TableStructureDetection.*;
import com.maheffa.TabulatedOCR.TextExtraction.*;
import org.w3c.dom.*;
import javax.swing.*;
import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.awt.image.BufferedImage;
import java.util.*;

/**
 * @author mahefa
 */
public class TOCRWorker extends SwingWorker<HashMap<String, Object>, RunnerProgress> {
    (...)

    TOCRWorker(Ocr MainForm mainForm) {
        (...)
    }

    @Override
    protected HashMap<String, Object> doInBackground() throws Exception {
        int nStep = 13;
        Ocrconfig conf = mainForm.getCurrentConfiguration();
        Format format = mainForm.getCurrentFormat();
        Project project = mainForm.getCurrentProject();
        String path = project.getInputFilePath();
        String[] pbe = ImgProcUtil.getPathBaseExtension(path);
        CellContainer cellContainer = null;

        System.out.println("Configuration:\n" + conf);

        mainForm.showProgress();
        // STATE_READING_IMAGE
        publish(RunnerProgress.createMessenger(0, "Reading image from " + path));
    }
}
```

Изм.	Лист	№ докум.	Подпись	Дат
Разраб.	Манитарию А.М.			
Руковод.	Брусенцев А.И.			
Консул.				
Н. контр.	Полунин А.И.			
Зав. каф.	Поляков В.М.			

ПРИЛОЖЕНИЕ Б
Программная реализация
алгоритмов на основе
искусственных нейронных сетей
для оцифровки табличных
структур с бумажных носителей

Лит.	Лист	Листов
	104	116
БГТУ им. В. Г. Шухова, ПВ-51		

```

        BufferedImage originalImage = ImgProcUtil.readImage(path);
        publish(RunnerProgress.createImageShower(RunnerProgress.STATE_READING_IMAGE,
        originalImage));

    // STATE_GRAYSCALING
        publish(RunnerProgress.createMessenger((int) (100.0 / nStep), "Grayscale ..."));
        BinaryImage binaryImage = new BinaryImage(originalImage);
        binaryImage.convertToGrayScale(conf.getGrayscale());
        publish(RunnerProgress.createImageShower(RunnerProgress.STATE_GRAYSCALING,
        binaryImage.rasterize()));

    // STATE_BINARISING
        BufferedImage workingImage = binaryImage.rasterize();
        publish(RunnerProgress.createMessenger((int) (100.0 * 2 / nStep), "Binarising ..."));
        binaryImage.binarize(conf.getBinarisation());
        publish(RunnerProgress.createImageShower(
            RunnerProgress.STATE_BINARISING,
            workingImage));

    // STATE_DETECTING_CONNECTED_PIXELS
        publish(RunnerProgress.createMessenger((int) (100.0 * 3 / nStep), "Detecting connected
pixel ..."));
        ArrayList<ConnectedPixel> connectedPixels = ConnectedPixel.getConnectedPixels(
            conf.getRadius(),
            conf.getMargin(),
            binaryImage
        );
        publish(RunnerProgress.createMessenger(-1, "Done connected pixel ..."));

        TableDetector tableDetector = null;
        BufferedImage largeTable = null;
        if (format.getType().equalsIgnoreCase("TABLE")) {
    // STATE_GETTING_LARGEST_CONNECTED_PIXELS
            tableDetector = new TableDetector(workingImage);
            tableDetector.setConfiguration(conf);
            publish(RunnerProgress.createMessenger((int) (100.0 * 4 / nStep), "Getting largest
connected pixel ..."));
            largeTable = tableDetector.getLargestTable(connectedPixels);
        }

    // STATE_APPLYING_HOUGH
        publish(RunnerProgress.createMessenger((int) (100.0 * 5 / nStep), "Applying Hough
transform ..."));
        TableDetector tmpTableDetector = new TableDetector(workingImage);
        tmpTableDetector.setConfiguration(conf);
        BufferedImage tmpHough = tmpTableDetector.applyHoughLineProbabilistic();
        double skew = tmpTableDetector.getDocumentSkew();

        publish(RunnerProgress.createImageShower(RunnerProgress.STATE_APPLYING_HOUGH,
        tmpHough));

```

Изм.	Лист	№ докум.	Подпись	Дата

```

if (conf.getDeskew()) {
    publish(RunnerProgress.createMessenger(-1, "Checking image skew"));
    if (skew < conf.getTolerableSkewAngle()) {
        publish(RunnerProgress.createMessenger(-1, "Document doesn't need deskewing"));
        System.out.println("Image doesn't need deskewing");
    } else {
// STATE_DESKEWING_IMAGE
        System.out.println("Deskewing image");
        publish(RunnerProgress.createMessenger((int) (100.0 * 6 / nStep), "Deskewing
image"));
        workingImage = tmpTableDetector.deskewDocument(workingImage);

publish(RunnerProgress.createImageShower(RunnerProgress.STATE_DESKEWING_IMAGE,
workingImage));
        binaryImage = new BinaryImage(workingImage, 100);
    }
}

if (format.getType().equalsIgnoreCase("TABLE")) {
    BufferedImage houghTable = tableDetector.applyHoughLineProbabilistic(largeTable);
// STATE_GETTING_LARGEST_TABLE
    publish(RunnerProgress.createMessenger((int) (100.0 * 9 / nStep), "Getting table"));
    connectedPixels = ConnectedPixel.getConnectedPixels(
        conf.getRadius(),
        conf.getMargin(),
        binaryImage
    );
    largeTable = tableDetector.getLargestTable(connectedPixels);
    tableDetector = new TableDetector(largeTable);
    tableDetector.setConfiguration(conf);
    tableDetector.applyHoughLineProbabilistic();

// STATE_GETTING_PERFECT_TABLE
    publish(RunnerProgress.createMessenger((int) (100.0 * 10 / nStep), "Getting perfect
table"));
    BufferedImage perfectTable = tableDetector.getLineApproximation()
        .draw(workingImage.getWidth(), workingImage.getHeight(),
        conf.getLineThickness());
    publish(RunnerProgress.createImageShower(RunnerProgress.STATE_GETTING_PERFECT_T
ABLE, perfectTable));

// STATE_CHECKING_CELL
    publish(RunnerProgress.createMessenger((int) (100.0 * 11 / nStep), "Checking cells"));
    CellExtractor cellExtractor = new CellExtractor(
        perfectTable,
        tableDetector.getIntersetionPoints(),
        conf.getLineThickness()
    );
    cellContainer = cellExtractor.getCellContainer();
    cellExtractor.drawOnImage(largeTable, cellContainer);
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

publish(RunnerProgress.createImageShower(RunnerProgress.STATE_CHECKING_CELL,
largeTable));
}

// STATE_RECOGNIZING_TEXT
publish(RunnerProgress.createMessenger((int) (100.0 * 12 / nStep), "Recognizing text"));
Extractor extractor = new Extractor();
extractor.setLanguage(conf.getLanguage());
extractor.setApplyNoiseRemoval(conf.getDenoise());
extractor.setUserDictionary(conf.getUserDictionary());
extractor.setTreatCell(format.getType().equalsIgnoreCase("TABLE"));
ArrayList<ArrayList<String>> cellText = null;
String text = null;
if (format.getType().equalsIgnoreCase("TABLE")) {
    BufferedImage bImg = binaryImage.rasterize();
    cellText = new ArrayList<ArrayList<String>>();
    cellContainer.initLines();
    while (cellContainer.hasNextLine()) {
        publish(RunnerProgress.createMessenger(
            (int) (100.0 * 12 / nStep + 100 * cellContainer.percentage() / nStep),
            "Recognizing text"));
        cellText.add(cellContainer.extractLine(extractor, bImg));
    }
} else {
    text = extractor.extractText(binaryImage.rasterize());
}
// STATE_GETTING_VARIABLES
publish(RunnerProgress.createMessenger((int) (100.0 * 13 / nStep), "Getting variables"));
HashMap<String, Object> variables = new HashMap<String, Object>();
if (format.getType().equalsIgnoreCase("TABLE")) {
    TableFormat tableFormat = format.getTableFormat();
    int columnCount = tableFormat.getColumnCount();
    ColumnCharacteristic[] columns = new ColumnCharacteristic[columnCount];
    for (int i = 0; i < columnCount; i++) {
        for (ColumnCharacteristic col : (Set<ColumnCharacteristic>)
            tableFormat.getColumnCharacteristics()) {
            if (col.getPosition() == i) {
                columns[i] = col;
                variables.put(col.getName(), new ArrayList<String>());
                break;
            }
        }
    }
    int index = tableFormat.getReadFirstLine() ? 0 : 1;
    for (; index < cellText.size(); index++) {
        ArrayList<String> line = cellText.get(index);
        if (line.size() < columnCount) continue;
        for (int i = 0; i < columns.length; i++) {
            ((ArrayList<String>) variables.get(columns[i].getName())).add(line.get(i));
        }
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```
        }
    } else {
        HashMap<String, String> tmpVariable = FuzzyTextMatcher.matchWholeVariables(
            text,
            format.getTextFormat().getContent(),
            this
        );
        for (Map.Entry<String, String> entry : tmpVariable.entrySet()) {
            variables.put(entry.getKey(), entry.getValue());
        }
    }
    publish(RunnerProgress.createMessenger(100, "Done"));
    return variables;
}
@Override
public void done() {
    ...
}
}
```

Изм.	Лист	№ докум.	Подпись	Дата

ПРИЛОЖЕНИЕ Б

Приложение В

Основные функции класса TableDetector

```
package com.maheffa.TabulatedOCR.TableStructureDetection;

import (...);

/**
 * @author mahefa
 */
public class TableDetector {

    private BufferedImage binaryImage = null;
    private String filePath = "";
    private int threshold1 = 20;
    private int threshold2 = 60;
    private double distanceAccumulator = 1;
    private double angleAccumulator = Math.PI / 180;
    private int thresholdAccumulator = 40;
    private int minimumLineLenght = 50;
    private int maximumLineGap = 100;
    private int minimumTableArea = 100 * 100;
    private double documentSkew = 0;
    private LineApproximation lineApproximation;
    private double toleranceAngle;

    (...)

    public BufferedImage applyHoughLineProbabilistic(BufferedImage image) {
        System.out.println("Starting hough probabilistic process");
        double sumLenght = 0;
        double sumMult = 0;
        System.out.println("reading on\n" + ImgProcUtil.getImageInfoHTML(image));
        Mat src = bufferedImageToMat(image);
        Mat dst = new Mat();
        Mat cdst = new Mat();
        Mat lines = new Mat();
        System.out.println("Canny");
        Canny(src, dst, threshold1, threshold2);
        System.out.println("Converting color");
        cvtColor(dst, cdst, COLOR_GRAY2BGR);
        System.out.println("HoughLinesP");
        HoughLinesP(dst, lines, distanceAccumulator, angleAccumulator, thresholdAccumulator,
                    minimumLineLenght, maximumLineGap);
        System.out.println("Detected " + lines.rows() + " lines");
        for (int i = 0; i < lines.rows(); i++) {
```

Изм.	Лист	№ докум.	Подпись	Дат
Разраб.	Манитариву А.М.			
Руковод.	Брусенцев А.И.			
Консул.				
Н. контр.	Полунин А.И.			
Зав. каф.	Поляков В.М.			

Программная реализация
алгоритмов на основе
искусственных нейронных сетей
для оцифровки табличных
структур с бумажных носителей

ПРИЛОЖЕНИЕ В

Лит.	Лист	Листов
	109	116
БГТУ им. В. Г. Шухова, ПВ-51		

```

        for (int j = 0; j < lines.cols(); j++) {
            double[] vec = lines.get(i, j);
            double x1 = vec[0],
                  y1 = vec[1],
                  x2 = vec[2],
                  y2 = vec[3];
            Point pt1 = new Point(x1, y1);
            Point pt2 = new Point(x2, y2);
            double angle = (180 / Math.PI * getAngle(pt1.x, pt1.y, pt2.x, pt2.y));
            double distance = getDistance(pt1.x, pt1.y, pt2.x, pt2.y);
            if (angle > 45) {
                angle -= 90;
            } else if (angle < -45) {
                angle += 90;
            }
            System.out.println("\t " + pt1 + " - " + pt2 + "; angle: " + (int) angle + ", distance: " +
(int) distance);
            line(cdst, pt1, pt2, new Scalar(255, 0, 0), 3);
            if (angle > 20 || angle < -20) continue;
            sumLenght += distance * distance;
            sumMult += angle * distance * distance;
            lineApproximation.add((int) x1, (int) y1, (int) x2, (int) y2);
        }
    }
    documentSkew = sumMult / sumLenght;
    System.out.println("Document skew : " + documentSkew);
    return matToBufferedImage(cdst);
}

public ArrayList<Point> getInterseptionPoints() {
    ArrayList<Point> intersections = new ArrayList<Point>();
    ArrayList<Line> lines = lineApproximation.getLines();
    for (int i = 0; i < lines.size(); i++) {
        for (int j = i + 1; j < lines.size(); j++) {
            Point p = Line.getProlongedInterseption(lines.get(i), lines.get(j));
            if (p.x > 0 && p.x < binaryImage.getWidth()
                && p.y > 0 && p.y < binaryImage.getHeight()) {
                intersections.add(p);
            }
        }
    }
    System.out.println("Number of interseption " + intersections.size());
    return intersections;
}

(...)

public double getDocumentSkew() {
    return Double.isNaN(documentSkew) || Double.isInfinite(documentSkew) ? 0 :
documentSkew;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```
public BufferedImage deskewDocument(BufferedImage img) {  
    return rotate(img, -getDocumentSkew() * Math.PI / 180);  
}  
  
(...)  
}
```

Изм.	Лист	№ докум.	Подпись	Дата

ПРИЛОЖЕНИЕ В

Лист
111

Приложение Г

Основные функции класса FuzzyTextMatcher

```
package com.maheffa.TabulatedOCR.TextExtraction;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * @author mahefa
 */
public class FuzzyTextMatcher {

    (...)

    public static int LevenshteinDistance(String str1, String str2) {
        int[][] distance = new int[str1.length() + 1][str2.length() + 1];

        for (int i = 0; i <= str1.length(); i++)
            distance[i][0] = i;
        for (int j = 1; j <= str2.length(); j++)
            distance[0][j] = j;

        for (int i = 1; i <= str1.length(); i++)
            for (int j = 1; j <= str2.length(); j++)
                distance[i][j] = minimum(
                    distance[i - 1][j] + 1,
                    distance[i][j - 1] + 1,
                    distance[i - 1][j - 1] + ((str1.charAt(i - 1) == str2.charAt(j - 1)) ? 0 : 1));

        return distance[str1.length()][str2.length()];
    }

    public static int LevenshteinDistance(char[] str1, char[] str2, int i0, int j0, int i1, int j1) {
        int[][] distance = new int[j0 - i0 + 1][j1 - i1 + 1];
        for (int i = 0; i <= j0 - i0; i++) {
            distance[i][0] = i;
        }
        for (int i = 0; i <= j1 - i1; i++) {
            distance[0][i] = i;
        }
        for (int i = 1; i <= j0 - i0; i++) {
            for (int j = 1; j <= j1 - i1; j++) {
                distance[i][j] = minimum(
                    distance[i - 1][j] + 1,
                    distance[i][j - 1] + 1,
                    distance[i - 1][j - 1] + ((str1[i0 + i - 1] == str2[j0 + j - 1]) ? 0 : 1));
            }
        }
    }
}
```

Изм.	Лист	№ докум.	Подпись	Дат	ПРИЛОЖЕНИЕ Г			
Разраб.	Манитариву А.М.				Программная реализация алгоритмов на основе искусственных нейронных сетей для оцифровки табличных структур с бумажных носителей	Лим.	Лист	Листов
Руковод.	Брусенцев А.И.						112	116
Консул.								
Н. контр.	Полунин А.И.							
Зав. каф.	Поляков В.М.							
БГТУ им. В. Г. Шухова, ПВ-51								

```

        distance[i][j] = minimum(
            distance[i - 1][j] + 1,
            distance[i][j - 1] + 1,
            distance[i - 1][j - 1] + ((str1[i + i0 - 1] == str2[j + i1 - 1]) ? 0 : 1)
        );
    }
    return distance[j0 - i0][j1 - i1];
}

(...)

public static int[] substringMatch(String str, String match) {
    // return [beg, end] where str[beg - end] has the minimum distance to match
    int minBeg = 0;
    int minEnd = 0;
    int minDist = Integer.MAX_VALUE;
    char[] str1 = str.toCharArray();
    char[] str2 = match.toCharArray();
    for (int beg = 0; beg < str.length(); beg++) {
        for (int end = Math.min(str.length(), beg + (int) (0.7 * match.length())); end <=
Math.min(str.length(), beg + (int) (1.3 * match.length())); end++) {
            int val = LevenshteinDistance(str1, str2, beg, end, 0, str2.length());
            if (val < minDist) {
                minBeg = beg;
                minEnd = end;
                minDist = val;
            }
        }
    }
    return new int[]{minBeg, minEnd};
}

(...)

public static ArrayList<int[]> getBeginEndOfVariables(String formatted) {
    ArrayList<int[]> beginEnds = new ArrayList<int[]>();
    Pattern pattern = Pattern.compile("(\\$[a-zA-Z_0-9]+)");
    Matcher matcher = pattern.matcher(formatted);
    while (matcher.find()) {
        beginEnds.add(new int[]{matcher.start(), matcher.end()});
    }
    return beginEnds;
}

public static HashMap<String, String> matchWholeVariables(String text, String format,
TOCRWorker worker) {
    System.out.println("matching variables");
    System.out.println("Input:\n" + text);
    System.out.println("Format:\n" + format);
    HashMap<String, String> map = new HashMap<String, String>();
}

```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

```

ArrayList<int[]> beginEndOfVariables = getBeginEndOfVariables(format);
int pt = 0;
for (int i = 0; i < beginEndOfVariables.size(); i++) {
    boolean last = i == beginEndOfVariables.size() - 1;
    int[] beVar0 = beginEndOfVariables.get(i);
    int[] beVar1;
    if (!last) {
        beVar1 = beginEndOfVariables.get(i + 1);
    } else {
        beVar1 = new int[]{format.length(), format.length()};
    }
    System.out.println("Finding value of variable " + format.substring(beVar0[0],
beVar0[1]));
    if (worker != null) {
        worker.forcePublish(RunnerProgress.createMessenger(
            (int) (100 * 12.0 / 13 + 100.0 * i / 13 / beginEndOfVariables.size()),
            "Reading value of " + format.substring(beVar0[0], beVar0[1])));
    }
    int[] txt0 = substringMatch(text, format.substring(pt, beVar0[0]));
    txt0[1]--;
    int[] txt1 = substringMatch(text, format.substring(
        beVar0[1],
        last ? format.length() : beVar1[0]));
    txt1[1]--;
    getExtendedBegEnd(text, txt0);
    getExtendedBegEnd(text, txt1);
    int a = txt0[1];
    int b = txt1[0] < a ? text.length() : txt1[0];
    map.put(
        getVarIn(format, beVar0[0]+1, beVar0[1]),
        cut(getVarIn(text, a, b)))
    );
    pt = beVar0[1];
}
System.out.println("Matched variables: ");
for (Map.Entry<String, String> entry : map.entrySet()) {
    System.out.println("\t" + entry.getKey() + " : " + entry.getValue());
}
return map;
}

public static String cut(String text) {
    int p = text.indexOf('\n');
    return text.substring(0, p <= 0 ? text.length() : p);
}

public static String cut(String text, int a, int b) {
    StringBuilder str = new StringBuilder();
    for (int i = a; i < b; i++) {
        if (text.charAt(i) == '\n') {
            break;
        }
    }
}
```

Изм.	Лист	№ докум.	Подпись	Дата

```

        } else {
            str.append(text.charAt(i));
        }
    }
    return str.toString();
}

public static String getVarIn(String text, int a, int b) {
    StringBuilder variable = new StringBuilder();
    boolean gotSpace = true;
    while (a < b && a < text.length()) {
        char s = text.charAt(a);
        if (gotSpace) {
            if (s == ' ') {
                a++;
            } else {
                variable.append(s);
                a++;
                gotSpace = false;
            }
        } else {
            if (s == '\n') {
                break;
            } else if (s == ' ') {
                gotSpace = true;
            }
            variable.append(s);
            a++;
        }
    }
    if (gotSpace && variable.length() > 0) {
        variable.deleteCharAt(variable.length() - 1);
    }
    return variable.toString();
}

public static String clean(String str) {
    return str.replaceAll("\\s+", " ");
}

public static void getExtendedBegEnd(String text, int[] beginEnd) {
    int begin = beginEnd[0];
    int end = beginEnd[1];
    while (begin >= 0 && text.charAt(begin) != ' ' && text.charAt(begin) != '\n') {
        begin--;
    }
    begin++;
    while (end >= 0 && end < text.length() && text.charAt(end) != ' ' && text.charAt(end) != '\n') {
        end++;
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```
beginEnd[0] = begin;  
beginEnd[1] = end;  
}  
(...)  
}
```

Изм.	Лист	№ докум.	Подпись	Дата

ПРИЛОЖЕНИЕ Г

Лист
116