МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

02.03.03 МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ И АДМИНИСТРИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

БАКАЛАВРСКАЯ РАБОТА

на тему
«Автоматизированное выделение границ легочного (сосудистого) рисунка»

Студент Д.И. Краснов _____

Руководитель М.Г. Лисовская _____

Консультант по аннотации А.В. Кириллова _____

Допустить к защите
Заведующий кафедрой к.т.н. А.В. Очеповский _____

«_____» _____ 2017 г.

Тольятти 2017

АННОТАЦИЯ

Объектом исследования выпускной квалификационной работы является процесс выделения границ на рисунке с использованием языка технических расчетов MATLAB.

Целью данной работы является разработка программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка.

Разработано программное обеспечение для автоматизированного выделения границ легочного рисунка и поиска патологий.

Проведен анализ состояния алгоритмов выделения границ на изображении, рассмотрены существующие аналоги и язык программирования для реализации алгоритмов, сформулированы цель и задачи разработки, а также требования к алгоритмам.

Реализованы алгоритмы выделения границ легочного рисунка, поиска патологий на флюорографических снимках и проверки масок, описана работа данных алгоритмов.

Произведено тестирование каждого из алгоритмов программного обеспечения с помощью пользовательского интерфейса.

Область применения ограничена городскими больницами и поликлиниками. Рассматривается вопрос о внедрении.

Работа состоит из введения, трех глав и заключения. Объем работы составляет 55 страниц, на которых размещены 34 рисунка, 3 таблицы и 3 приложения. При написании данной работы было использовано 30 литературных источников, в том числе 5 источников на иностранном языке.

ABSTRACT

The title of the graduation work is «Automated Border Selection in the Pulmonary (Vascular) Image». This graduation work is devoted to system software development for automated pathology search and border selection of the lungs in a fluorography image.

The readers' attention is drawn to significant increase in patients in the Russian Federation, who need the examinations with the help of fluorography. So the actual task is to increase the fluorographic examinations speed with the help of partial automation of the images analysis process.

The graduation work may be divided into several logically connected parts which are analysis of possible ways for pathology search and border selection of the lungs in the fluorography image, development of a program for the automated border selection of pathology and software testing.

The author dwells on the implementation of the pathologies searching, which is based on existing templates of the pathologies (masks). The graduation work describes in details the search for a mask inside the fluorography image. The special part of the graduation work gives details about setting an allowable error when searching for a mask.

The work is of interest for a wide circle of readers. The developed software is self-learning, but it has a low processing because of a large number of masks. It is compensated by the high accuracy of searching for pathology in an image.

The graduation work consists of an explanatory note on 55 pages, introduction, including 34 figures, 3 tables, the list of 30 references including 5 foreign sources and 3 appendices.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Туберкулез является одной из 10 ведущих причин смерти в мире. В 2015 году туберкулезом заболели 10,4 миллиона человек, и 1,8 миллиона человек умерли от этой болезни. Туберкулез является одной из ведущих причин смерти людей с ВИЧ: в 2015 году туберкулезом было вызвано 35% случаев смерти среди ВИЧ-инфицированных людей.

Также в 2015 году примерно у 480 000 людей в мире развился МЛУ-ТБ (туберкулез с множественной лекарственной устойчивостью). Кроме того, примерно 100 000 человек приобрели устойчивость к рифампицину (самому эффективному препарату первой линии) и нуждались в лечении МЛУ-ТБ. Наибольшим бременем проблема МЛУ-ТБ ложится на три страны – Китай, Индию и Российскую Федерацию, – на долю которых в совокупности приходится почти половина всех случаев в мире.

С 2000 года заболеваемость туберкулезом снижалась в среднем на 1,5% в год. Для достижения контрольных показателей на 2020 год, предусмотренных Стратегией по ликвидации туберкулеза, эти темпы снижения необходимо ускорить до 4–5% в год. По оценкам, за период с 2000 по 2015 год благодаря диагностике и лечению туберкулеза было спасено около 49 миллионов человеческих жизней. Одна из задач в области здравоохранения в рамках недавно принятых Целей в области устойчивого развития заключается в том, чтобы к 2030 году покончить с эпидемией туберкулеза [22].

На территории Российской Федерации многие пациенты нуждаются в проведении различного рода обследований при помощи рентгенографии. Рентгенологическое исследование – это применение рентгеновского излучения в медицине для изучения строения и функций различных органов и систем и распознавания заболеваний [23].

В современной рентгенологии существуют общие методы исследования, специальные и вспомогательные. К общим методам относят такие как

компьютерная томография, магнитно-резонансная томография, флюорография и другие.

В основе компьютерной томографии лежит специфическое свойство рентгеновского излучения поглощаться в зависимости от плотности конкретных тканей организма. На исследуемую область во время томографического рентгеновского исследования послойно воздействует пучок рентгеновских лучей, который, проходя сквозь ткани пациента с различной плотностью, поглощается ими. При этом возникают послойные изображения срезов тела. Высококачественное компьютерное оборудование обрабатывает полученные данные и перерабатывает их, создавая информативные трехмерные изображения, отражающие особенности исследуемого органа или участка тела.

Однако компьютерная томография оказывает значительное лучевое воздействие, поэтому не может применяться многократно.

В магнитно-резонансной томографии данные получают с применением мощного магнитного поля (ядерно-магнитного резонанса), благодаря которому атомы водорода в организме человека начинают менять свое положение. Томограф посылает электромагнитные импульсы, а возникающий в организме эффект улавливается оборудованием и перерабатывается в трехмерные изображения.

Рентгеновские лучи при проведении компьютерной томографии воздействуют на органы и ткани до 10 секунд, что предпочтительно для лиц страдающих клаустрофобией, а для полноценного МРТ-исследования может потребоваться 10–20 минут (с сохранением неподвижного состояния), поэтому при его проведении в детском возрасте часто применяют наркоз.

Магнитно-резонансная томография – один из наиболее точных методов диагностики широкого спектра патологий. Будучи безболезненной и совершенно безвредной для пациента методикой, МРТ в настоящее время широко применяется для выявления различных проблем с тканями и органами. Однако при всех неоспоримых достоинствах у МРТ есть и недостатки, а именно – не слишком достоверная визуализация полых органов, к которым

относятся в числе прочих и легкие. Поэтому зачастую для подтверждения выявленной на МРТ-снимке патологии назначается флюорография.

Флюорография – это рентгенологическое исследование, заключающееся в фотографировании видимого изображения на флуоресцентном экране, которое образуется в результате прохождения рентгеновских лучей через тело человека и неравномерного поглощения органами и тканями организма. Наиболее распространенным диагностическим методом, использующим принцип флюорографии, является флюорография органов грудной клетки, которая применяется прежде всего для скрининга туберкулеза и новообразований легких. Флюорография особенно ценна тем, что позволяет выявлять даже те патологии, которые протекают бессимптомно.

Основные преимущества флюорографии перед другими методами рентгенологического исследования – большая пропускная способность при малой затрате времени на каждое исследование, возможность приблизить обследование к населению, экономичность, относительно небольшая профессиональная вредность [24].

Для подобных рентгенологических исследований были разработаны как стационарные, так и мобильные флюорографические аппараты. При этом в комплект программного обеспечения для флюорографических аппаратов, приобретаемых российскими медицинскими учреждениями, не входят программы, предназначенные для анализа снимков (обнаружения образований) флюорографических аппаратов.

Ввиду значительного увеличения числа больных на территории Российской Федерации, нуждающихся в проведении различного рода обследований при помощи рентгенографии, актуальной задачей является повышение скорости проведения подобных обследований путем частичной автоматизации процесса анализа снимков.

Исходя из актуальности рассматриваемой темы была поставлена цель и определены объект и предмет выпускной квалификационной работы.

Цель выпускной квалификационной работы: разработка программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка.

Объект исследования выпускной квалификационной работы: процесс выделения границ на рисунке с использованием языка технических расчетов MATLAB.

Предмет исследования выпускной квалификационной работы: автоматизация процесса выделения границ легочного (сосудистого) рисунка.

В ходе выполнения выпускной квалификационной работы должно быть разработано программное обеспечение для автоматизированного выделения границ легочного (сосудистого) рисунка.

Выпускная квалификационная работа состоит из введения, трех глав и заключения.

Во введении описывается актуальность рассматриваемой темы, определяются объект и предмет выпускной квалификационной работы, ставится цель и выявляются задачи.

В первой главе происходит анализ состояния вопроса, а именно обзор существующих программных аналогов, формализация требований к разрабатываемому программному продукту и описание использованного языка программирования для разработки программного обеспечения.

Во второй главе описывается разработка алгоритмов программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка.

В третьей главе происходит тестирование разработанного программного обеспечения в соответствии с формализованными требованиями к программному продукту.

В заключении подводятся итоги разработки программного обеспечения, формируются окончательные выводы по рассматриваемой теме.

1 АНАЛИЗ СОСТОЯНИЯ АЛГОРИТМОВ ВЫДЕЛЕНИЯ ГРАНИЦ НА ИЗОБРАЖЕНИИ

1.1 Обзор программных аналогов выделения границ на изображении

В Национальном исследовательском Томском политехническом университете Е.И. Максимовой при выполнении бакалаврской работы был разработан алгоритм обнаружения образований в легких человека на снимках компьютерного томографа с использованием искусственной нейронной сети. Предложенный метод состоит из нескольких ключевых этапов:

- бинаризация – перевод изображения компьютерного томографа в монохромное изображение для упрощения процесса кластеризации;
- кластеризация – разделение бинаризированного изображения на белые и черные кластеры для определения положения легочных долей и последующей классификации кластеров внутри легочных долей;
- классификация – отнесение того или иного кластера к элементу легочного рисунка или к отклонению от нормы.

В данной работе был использован алгоритм адаптивной бинаризации, реализованный в виде консольного приложения на языке программирования C++ с использованием библиотеки компьютерного зрения с открытым исходным кодом OpenCV.

Пример результата бинаризации изображения здорового легкого представлен на рисунке 1.1.

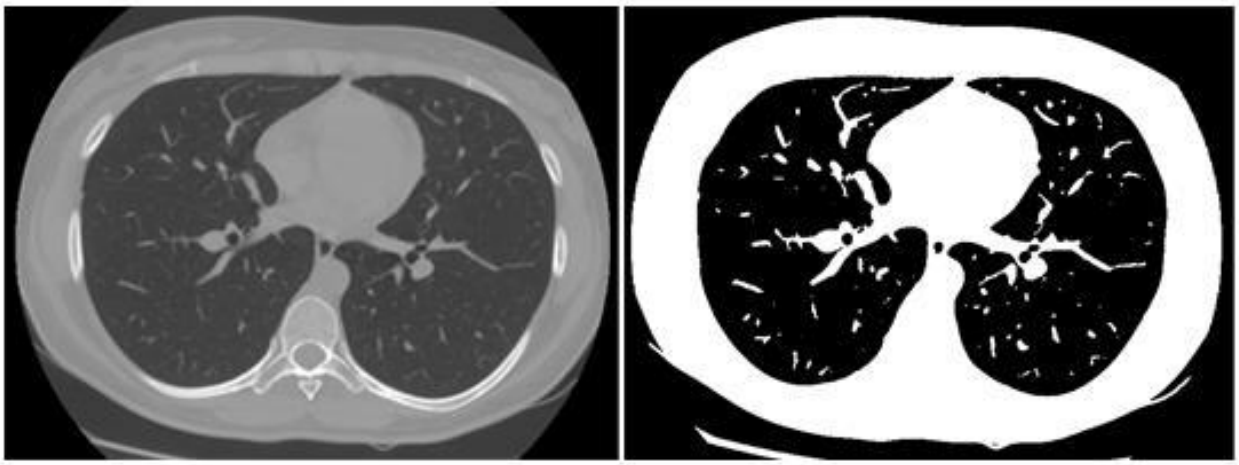

Рисунок 1.1 – Изображение здорового легкого и результат его бинаризации

Для реализации алгоритма разбиения бинаризированного изображения на кластеры использовался волновой алгоритм (алгоритм обхода в ширину). Внутри черного кластера, соответствующего легочной доле, проводится поиск всех белых кластеров, среди которых могут быть как образования в легких, так и элементы легочного рисунка. Определяется тип каждого из таких кластеров: элемент легочного рисунка, образование в легких, легочная жидкость и другие.

Для реализации алгоритма классификации была использована искусственная нейронная сеть. Для ее обучения использовался алгоритм обратного распространения ошибки. Было отмечено, что с ростом количества нейронов в скрытом слое растет число итераций, необходимых для обучения искусственной нейронной сети, при этом результаты работы алгоритма улучшаются несущественно.

Результатом работы программы является изображение, аналогичное исходному, на котором синим цветом выделены объекты, распознанные как образования в легких.

Пример результата работы программы обнаружения образований в легких человека на снимках компьютерного томографа представлен на рисунке 1.2.

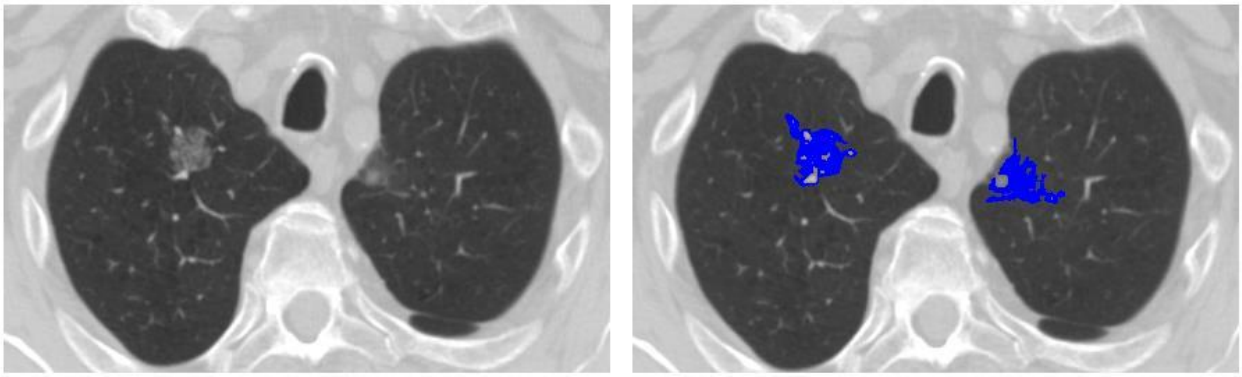

Рисунок 1.2 – Пример результата работы программы обнаружения образований

Е.И. Максимова подчеркивает, что вычислительная мощность алгоритма линейно зависит от количества пикселей на изображении, что говорит о возможности применения такого алгоритма в системах, обладающих высоким быстродействием [14].

1.2 Анализ языка технических расчетов MATLAB

MATLAB – это высокоуровневый язык и интерактивная среда для программирования, численных расчетов и визуализации результатов. С помощью MATLAB можно анализировать данные, разрабатывать алгоритмы, создавать модели и приложения.

Язык, инструментарий и встроенные математические функции позволяют пользователю исследовать различные подходы и получать решение быстрее, чем с использованием электронных таблиц или традиционных языков программирования, таких как C/C++ или Java.

MATLAB по сравнению с традиционными языками программирования позволяет на порядок сократить время решения типовых задач и значительно упрощает разработку новых алгоритмов.

MATLAB представляет собой основу всего семейства продуктов MathWorks и является главным инструментом для решения широкого спектра научных и прикладных задач, в таких областях как моделирование объектов и разработка систем управления, проектирование коммуникационных систем, обработка сигналов и изображений, измерение сигналов и тестирование, финансовое моделирование, вычислительная биология и других.

Ядро MATLAB позволяет максимально просто работать с матрицами реальных, комплексных и аналитических типов данных и со структурами данных и таблицами поиска.

MATLAB содержит встроенные функции линейной алгебры, быстрого преобразования Фурье, функции для работы с полиномами, функции базовой статистики и численного решения дифференциальных уравнений, а также расширенные математические библиотеки для Intel MKL.

Все встроенные функции ядра MATLAB разработаны и оптимизированы специалистами и работают быстрее их эквивалентов на C/C++ или Java [25], поэтому данный язык программирования поможет в эффективной реализации алгоритмов выделения границ на изображении.

1.3 Формализация требований и постановка задач к разрабатываемому программному обеспечению

Программное обеспечение для выделения границ легочного (сосудистого) рисунка должно содержать следующие алгоритмы:

- выделение границ легочного рисунка;
- поиск патологий на флюорографических снимках;
- проверка масок.

Алгоритм выделения границ легочного рисунка должен выделять на флюорографическом снимке область, соответствующую легочным долям. Элементы флюорографического снимка, находящиеся за пределами области легких, а также фон изображения, должны быть окрашены в белый цвет.

Алгоритм поиска патологий должен работать следующим образом:

- в программу должен быть загружен массив цветов полутонового (черно-белого) изображения – флюорографический снимок;
- программа должна произвести выделение легочной области на загруженном флюорографическом снимке;
- в программу последовательно должны быть загружены массивы цветов шаблонов с патологиями (маски) для проверки исходного изображения на наличие патологий;
- программа должна произвести поиск патологии внутри флюорографического снимка.

Поиск патологии должен осуществляться с заданным отличием (погрешностью) цветов между шаблоном маски с патологией и набором пикселей исходного изображения. Также внутри программы должен быть задан допустимый процент не совпадающих с маской элементов.

При обнаружении сходства между маской и набором пикселей исходного изображения программа должна создать новый шаблон маски с патологией на основе совпадающих с маской элементов и добавить его к уже имеющимся. Также программа должна выделить область найденного усиления и создать изображение с выделенными границами патологии.

Если патология не обнаружена с применением первого шаблона маски с патологией, программа должна перейти к следующему. Исходное изображение будет считаться нормой, если программа не обнаружит патологий на основе всех имеющихся шаблонов.

Алгоритм проверки масок должен проверять шаблоны масок с патологиями на их соответствие эталону. Также алгоритм должен применяться при добавлении нового шаблона маски с патологией к уже имеющимся.

Исходя из объекта и предмета исследования и учитывая особенности существующих аналогов, на основе формализованных требований к разрабатываемому программному обеспечению для достижения поставленной цели были определены следующие задачи:

1) изучить методы автоматизированного выделения границ на изображении;

2) разработать архитектуру программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка;

3) разработать программный код программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка с использованием языка технических расчетов MATLAB;

4) протестировать работу программного обеспечения для автоматизированного выделения границ легочного (сосудистого) рисунка.

В ходе выполнения выпускной квалификационной работы должно быть разработано программное обеспечение для автоматизированного выделения границ легочного (сосудистого) рисунка.

Произведены анализы программных аналогов и языка программирования программного обеспечения для выделения границ на изображении. Формализованы требования к алгоритмам и определены задачи для достижения поставленной цели.

2 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗИРОВАННОГО ВЫДЕЛЕНИЯ ГРАНИЦ ЛЕГОЧНОГО (СОСУДИСТОГО) РИСУНКА

2.1 Реализация алгоритма выделения границ легочного рисунка

Выделение границ легочного рисунка реализовано в несколько этапов.

Блок-схема алгоритма выделения границ легочного рисунка представлена на рисунке 2.1.

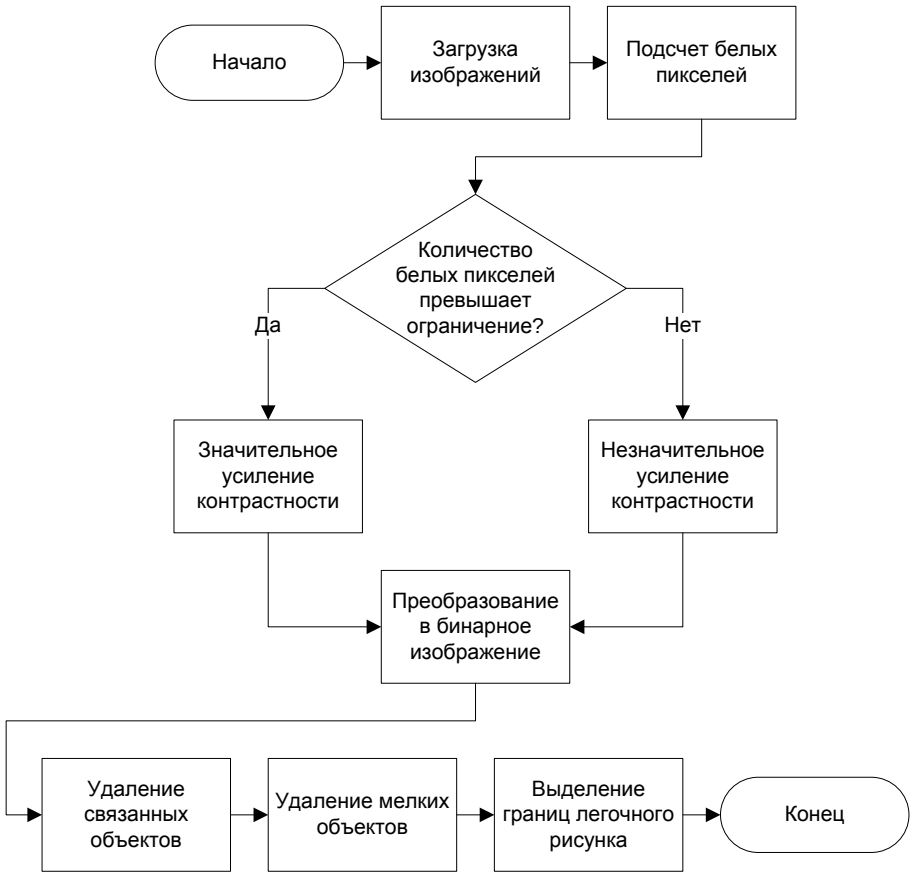

Рисунок 2.1 – Блок-схема алгоритма выделения границ легочного рисунка

Сначала в программу загружается исходный флюорографический снимок. Загрузка изображений из директории «Images» реализована с помощью команды dir, которая позволяет из командной строки системы MATLAB выполнить ряд команд DOS, связанных с управлением файлами. В переменную directory записываются данные о файлах с расширением, указанным в

переменной extension (в данном случае расширение .jpg), которые находятся в указанной директории (в данном случае директория «Images»).

Затем составляется одномерный массив Images с именами изображений из директории «Images». Поле .name отвечает за отделение имени файла от остальных данных, что позволяет записать в массив только имена изображений.

После вышеописанных действий запускается цикл загрузки массива изображений из директории «Images». Текущее изображение указывается с помощью добавления имени файла из массива Images к директории «Images» и загружается в массив Array.

Код загрузки изображений из директории «Images» приведен на рисунке 2.2.

```
extension = ('jpg');
directory = dir(['.\Images\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[mImages,nImages] = size(Images);
for countImages = 1 : nImages
    Array = imread(['.\Images\' Images{1,countImages}]);
end
```

Рисунок 2.2 – Код загрузки изображений из директории «Images»

Флюорографические изображения имеют два вида: вид спереди и вид сбоку. Изображения с видом сбоку затемнены внутри области легочного рисунка, но имеют большее количество пикселей с оттенками белого цвета за пределами этой области в отличие от изображений с видом спереди. Поэтому для выделения границ легочного рисунка необходимо увеличить контрастность изображения в соответствии с его видом для последующего преобразования в бинарное (черно-белое) изображение.

Учитывая, что все флюорографические снимки являются полутоновыми изображениями (256 оттенков цвета), значение оттенков белого цвета было установлено для пикселей с оттенками выше 240.

Гистограмма цветов флюорографических снимков для вида спереди и вида сбоку представлена на рисунке 2.3. На данной гистограмме можно заметить, что количество оттенков цветов белых пикселей на изображении вида

сбоку преобладает над количеством аналогичных оттенков на изображении вида спереди.

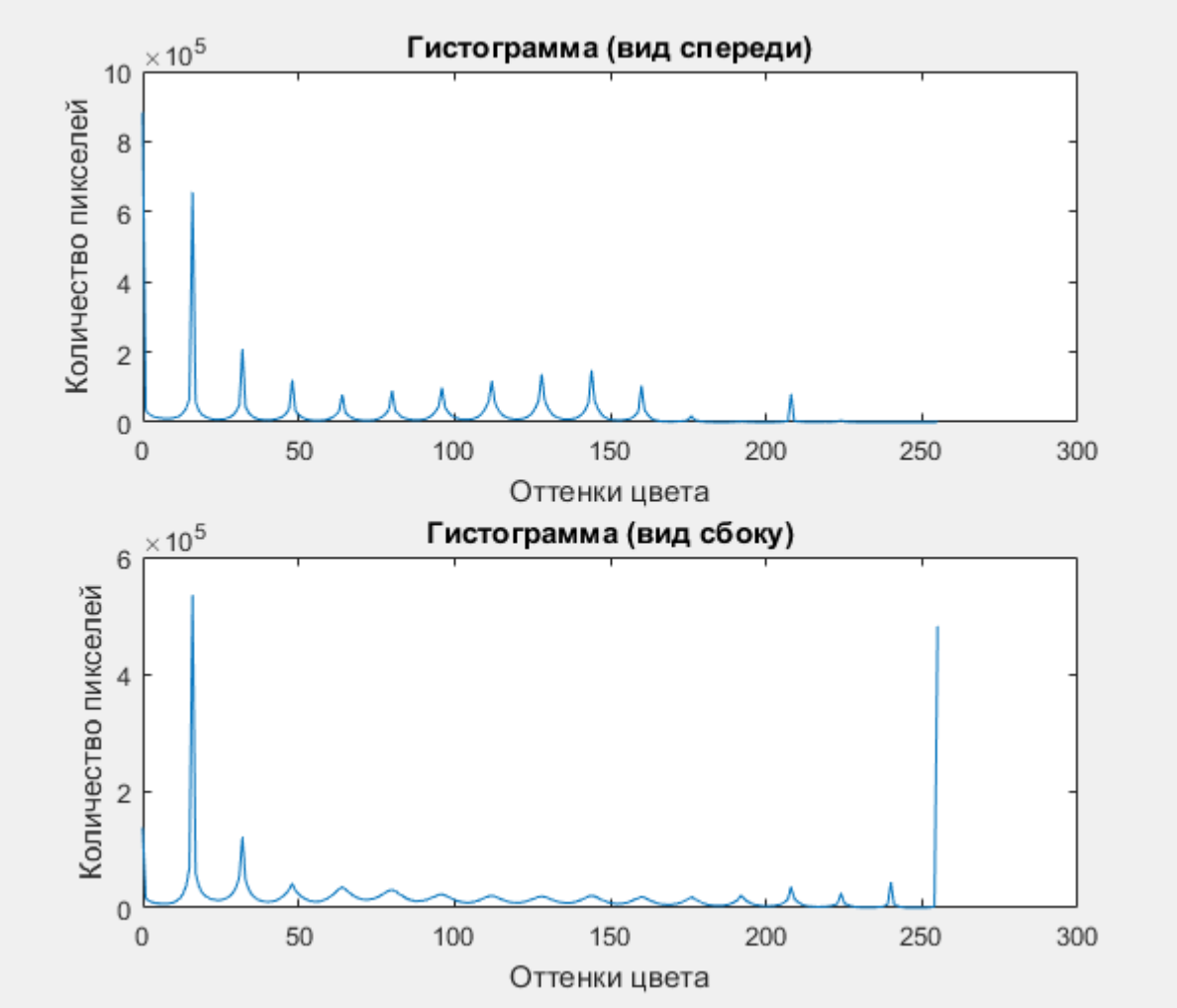

Рисунок 2.3 – Гистограмма оттенков цвета для вида спереди и вида сзади

Отношение количества пикселей белого цвета к размерности изображения вычисляется по формуле:

$$k = \frac{w}{n * m}, \tag{2.1}$$

где k – отношение количества белых пикселей к размерности изображения;

w – пиксели с оттенком цвета выше 240, пикс;

n – ширина изображения, пикс;

m – высота изображения, пикс.

Отношение количества белых пикселей к размерности флюорографического снимка для вида сбоку на основе формулы (2.1) представлено в таблице 2.1.

Таблица 2.1 – Отношение количества белых пикселей к размерности флюорографического снимка (вид сбоку)

Изображение	Ширина, пикс	Высота, пикс	Количество белых пикселей	Отношение
N5836DM-1.di.jpg	2533	1993	57372	0.011365
N5843DF-1.di.jpg	2533	1678	168339	0.039606
N5977DM-1.di.jpg	2298	2107	136118	0.028113
P5695D-1.di.jpg	2533	2450	101336	0.016329
P5970D-1.di.jpg	2533	2056	153921	0.029556

Отношение количества белых пикселей к размерности флюорографического снимка для вида спереди на основе формулы (2.1) представлено в таблице 2.2.

Таблица 2.2 – Отношение количества белых пикселей к размерности флюорографического снимка (вид спереди)

Изображение	Ширина, пикс	Высота, пикс	Количество белых пикселей	Отношение
N5843DF.di.jpg	2533	2415	5844	0.00095534
N5812DM.di.jpg	2533	2588	2006	0.00030601
NP5856D.di.jpg	2533	2405	5145	0.00084457
P5755D.di.jpg	2425	2356	2014	0.00035251
P5970D.di.jpg	2533	2723	2004	0.00029055

На основе полученных данных, представленных на рисунке 2.3 и в таблицах 2.1 и 2.2, можно сделать вывод, что флюорографические снимки с видом спереди имеют не больше 0,001% пикселей с оттенками белого цвета, в то время как снимки с видом сбоку имеют выше 0,01% подобных пикселей. Поэтому в алгоритме было установлено ограничение на поиск пикселей с оттенками белого цвета в 0,001% от всех пикселей на флюорографическом снимке.

Ограничение на поиск количества пикселей белого цвета вычисляется по формуле:

$$L = n * m * p, \tag{2.2}$$

где L – ограничение количества пикселей белого цвета, пикс;

n – ширина изображения, пикс;

m – высота изображения, пикс;

p – процент ограничения пикселей.

Переменная white является счетчиком количества пикселей с оттенками белого цвета. Переменная limit используется для ограничения поиска пикселей с оттенками белого цвета, равного 0,001% от всех пикселей изображения. После нахождения значения переменной limit запускается цикл подсчета количества пикселей с оттенками белого цвета в текущем изображении. Если количество подобных пикселей превышает значение limit, происходит выход из цикла.

Код поиска пикселей с оттенками белого цвета приведен на рисунке 2.4.

```
1     white = 0;
2
3     [rowArray,columnArray] = size(Array);
4     limit = round(rowArray * columnArray * 0.001);
5   for i = 1 : rowArray
6       if (white > limit)
7           break
8       end
9       for j = 1 : columnArray
10          if (white > limit)
11              break
12          end
13          if (Array(i,j) > 240)
14              white = white + 1;
15          end
16      end
17  end
```

Рисунок 2.4 – Код поиска пикселей с оттенками белого цвета

Для корректного преобразования в бинарное изображение необходимо увеличить контрастность флюорографического снимка в зависимости от количества уже имеющихся пикселей с оттенками белого цвета. Если количество таких пикселей превышает установленное ограничение limit (вид

сбоку), изображение получает большую контрастность, если их количество меньше ограничения limit (вид спереди), изображение получает меньшую контрастность.

Функция imadjust создает новое полутоновое изображение путем контрастирования исходного полутонового изображения. Чем меньше будет установлено второе значение в квадратных скобках внутри функции imadjust (от 0 до 255), тем более контрастным получится новое изображение.

Код усиления контрастности изображения в зависимости от количества пикселей с оттенками белого цвета приведен на рисунке 2.5.

```
1 -         if (white > limit)
2 -             gamma=imadjust(Array, [0 100]/255);
3 -         else
4 -             gamma=imadjust(Array, [0 200]/255);
5 -         end
```

Рисунок 2.5 – Код усиления контрастности изображения

Сравнение степени контрастности флюорографических снимков после применения функции imadjust для вида сбоку и вида спереди представлено на рисунках 2.6 и 2.7 соответственно. Слева – исходные изображения, справа – контрастные изображения.

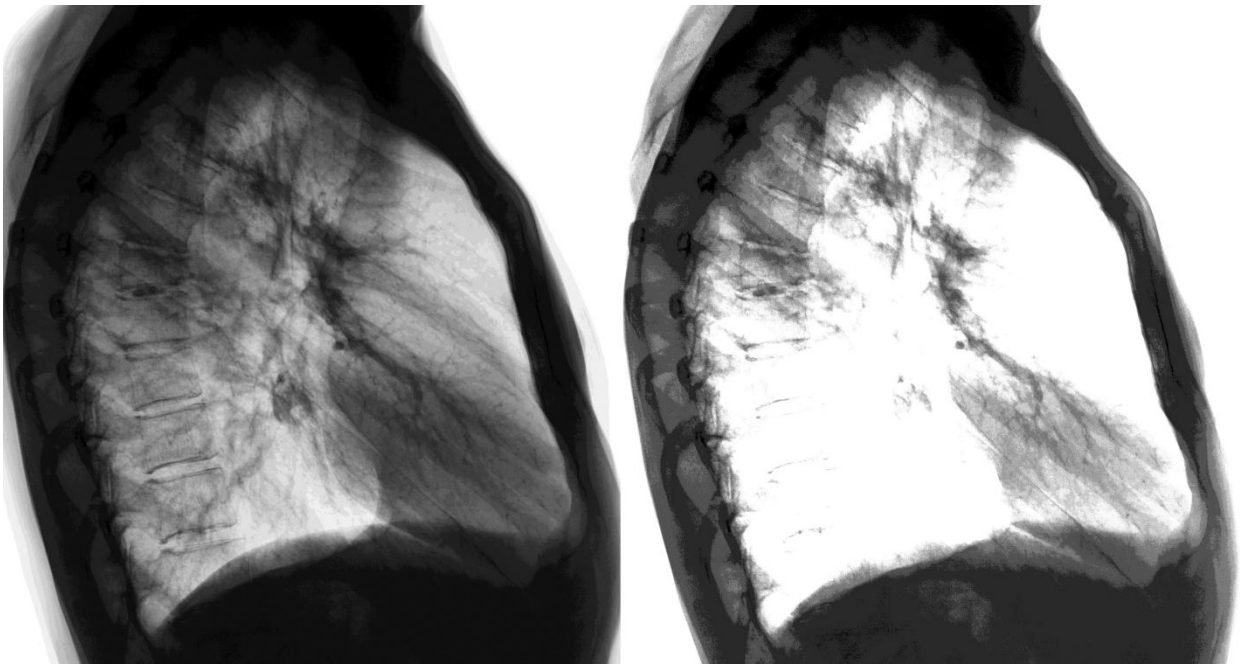

Рисунок 2.6 – Сравнение степени контрастности изображений (вид сбоку)

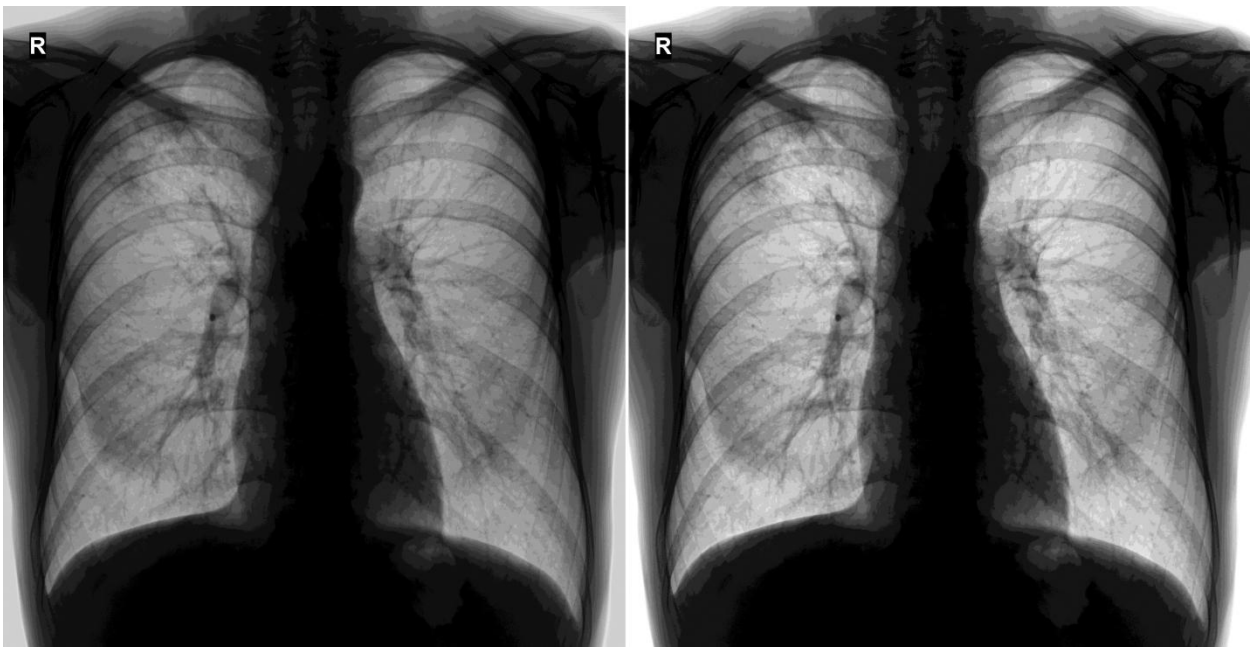

Рисунок 2.7 – Сравнение степени контрастности изображений (вид спереди)

После увеличения контрастности полутонового изображения происходит его преобразование в бинарное изображение с помощью функции im2bw.

Функция im2bw создает бинарное изображение, которое записывается в массив RibCage, используя отсечение по порогу яркости. Для этой цели полноцветные и палитровые изображения конвертируются в полутоновые. Пиксели результирующего бинарного изображения принимают значения 0 (черный цвет), если яркость соответствующих пикселей исходного изображения меньше глобального порога, и значения 1 (белый цвет), если яркость соответствующих пикселей исходного изображения больше или равна глобальному порогу.

Функция graythresh вычисляет значение глобального порога, который используется для преобразования яркостей (интенсивностей) изображения в бинарное изображение в функции im2bw. Часто данную операцию называют бинаризацией по порогу.

После преобразования флюорографического снимка в бинарное изображение необходимо убрать все лишние объекты. Объем таких объектов в текущем изображении вычисляется в переменной VObj, а значение объема в процентах задается в переменной VDel. Было установлено максимальное

значение объема для лишних объектов в 12.5% от общего количества пикселей. Функция bwareaopen удаляет с бинарного изображения все связные компоненты (объекты), площадь которых меньше значения в переменной VObj, и помещает результат обработки изображения в массив RibCage.

Функция imclose выполняет морфологическое закрытие полутонового или бинарного изображения, возвращая результат в массиве RibCage. Структурный элемент SE, в отличии от массива объектов, является единичным структурным элементом. Структура данного элемента задается функцией strel, тип элемента описывается в параметрах внутри функции (в данном случае тип элемента disk – круг, радиус – 5 пикселей). С помощью функции imclose происходит своего рода сглаживание границ изображения.

Также необходимо очистить бинарное изображение от возможных мелких объектов (например, точек и пятен). Функция imfill выполняет операцию заливки фоновыми пикселями (белыми) исходного изображения, начиная из точек, определенных параметром holes. Этот параметр определяет положение отверстий (точек) на исходном бинарном изображении.

Код преобразования контрастного флюорографического снимка в бинарное изображение приведен на рисунке 2.8.

```
VDel = 0.125;
VObj = round(rowArray * columnArray * VDel);

RibCage = im2bw(gamma, graythresh(gamma));
RibCage = bwareaopen(RibCage, VObj);
SE = strel('disk', 5);
RibCage = imclose(RibCage, SE);
RibCage=imfill(RibCage,'holes');
```

Рисунок 2.8 – Код преобразования контрастного флюорографического снимка в бинарное изображение

Сравнение исходных бинарных изображений и откорректированных с помощью функций bwareaopen, imclose и imfill для вида сбоку и вида спереди представлено на рисунках 2.9 и 2.10 соответственно. Слева – исходные бинарные изображения, справа – откорректированные изображения.

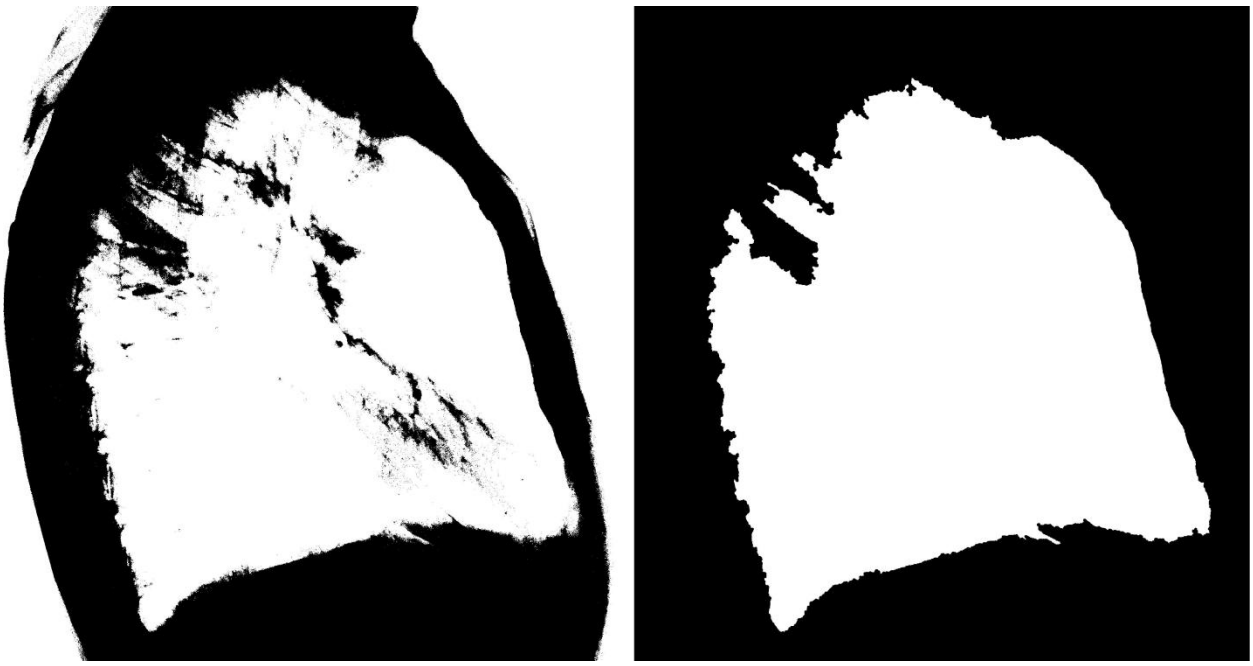

Рисунок 2.9 – Сравнение исходного бинарного изображения и откорректированного (вид сбоку)

Рисунок 2.10 – Сравнение исходного бинарного изображения и откорректированного (вид спереди)

В результате вышеописанных действий в массиве RibCage содержится бинарное изображение флюорографического снимка, на котором область легочного рисунка выделена белым цветом, а остальная область – черным.

Чтобы произвести выделение границ легочного рисунка, на исходном флюорографическом снимке необходимо окрасить белым цветом область, аналогичную области с пикселями черного цвета на бинарном изображении. Для этого запускается цикл поиска черных пикселей (значение цвета равно 0) по массиву с бинарным изображением RibCage. Так как исходный флюорографический снимок по размерности абсолютно идентичен полученному бинарному изображению, производится замена цветов для всех пикселей исходного изображения на белый (255 оттенок цвета), если аналогичный пиксель в бинарном изображении является черным.

Код замены цвета пикселей на исходном изображении в зависимости от значения цвета пикселей на бинарном изображении приведен на рисунке 2.11.

```
1    [rowArray,columnArray] = size(Array);
2    for i = 1 : rowArray
3        for j = 1 : columnArray
4            if (RibCage(i,j) == 0)
5                Array(i,j) = 255;
6            end
7        end
8    end
```

Рисунок 2.11 – Код замены цвета пикселей на исходном изображении

Сравнение исходных флюорографических снимков и полученных изображений с выделенными границами легочного рисунка для вида сбоку и вида спереди представлено на рисунках 2.12 и 2.13 соответственно. Слева – исходные флюорографические снимки, справа – изображения с выделенными границами легочного рисунка.

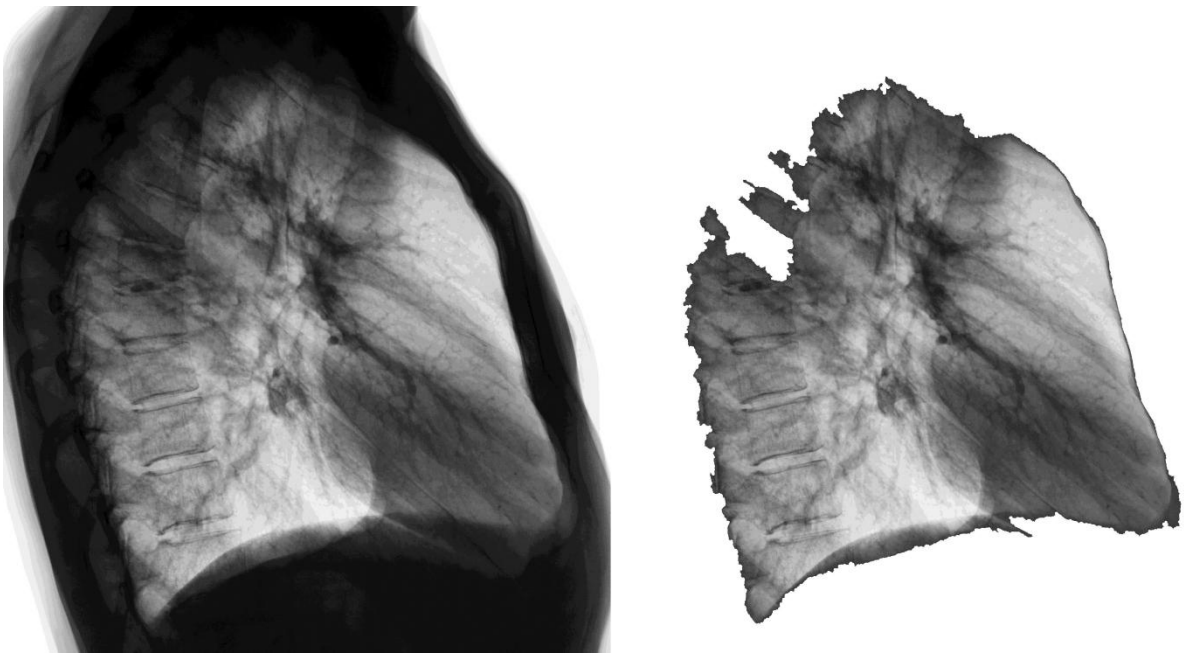

Рисунок 2.12 – Сравнение исходного флюорографического снимка и изображения с выделенными границами легочного рисунка (вид сбоку)

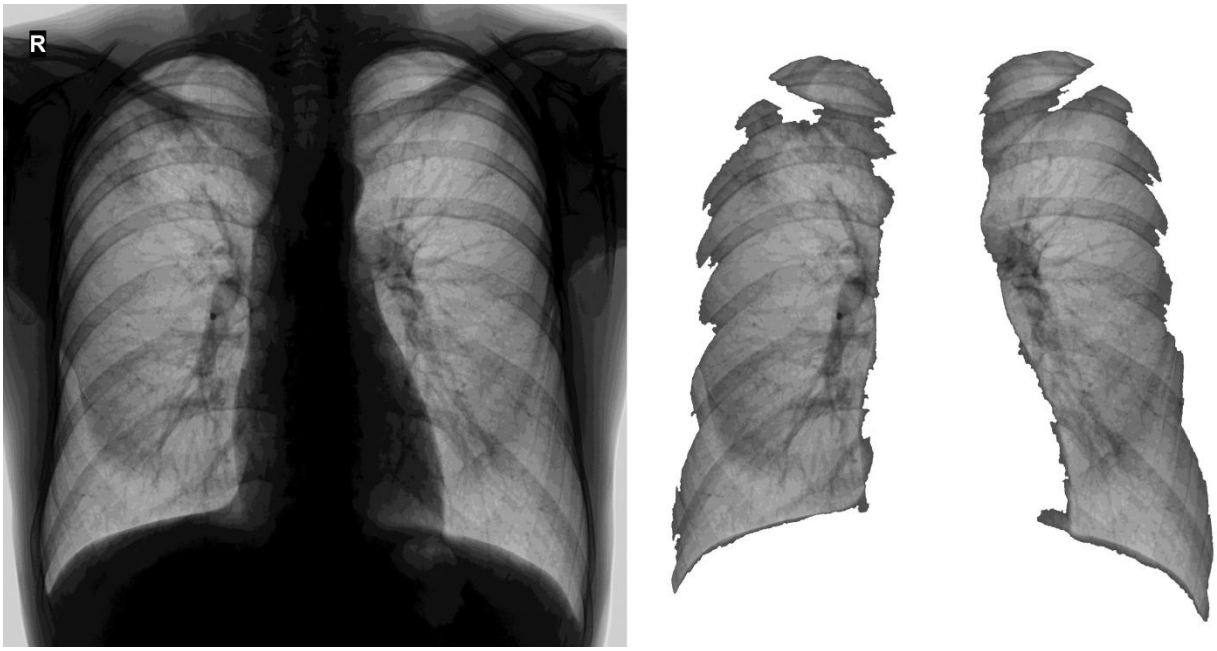

Рисунок 2.13 – Сравнение исходного флюорографического снимка и изображения с выделенными границами легочного рисунка (вид спереди)

Затем производится запись полученного изображения с выделенными границами легочного рисунка в директорию «Borders». К исходному имени файла добавляются символы «S_» (англ. selected – выбранный, выделенный),

которыми будут отмечаться все изображения с выделенными границами легочного рисунка.

Код алгоритма выделения границ легочного рисунка приведен в приложении А.

2.2 Реализация алгоритма поиска патологий на флюорографических снимках

Поиск патологий на флюорографических снимках реализован по принципу поиска маски с патологией внутри изображения. Любое изображение является массивом цветов пикселей, флюорографические снимки и маски с патологиями также не являются исключениями, поэтому поиск патологии с помощью маски – это своего рода поиск массива внутри массива.

Блок-схема алгоритма поиска патологий на флюорографических снимках представлена на рисунке 2.14.

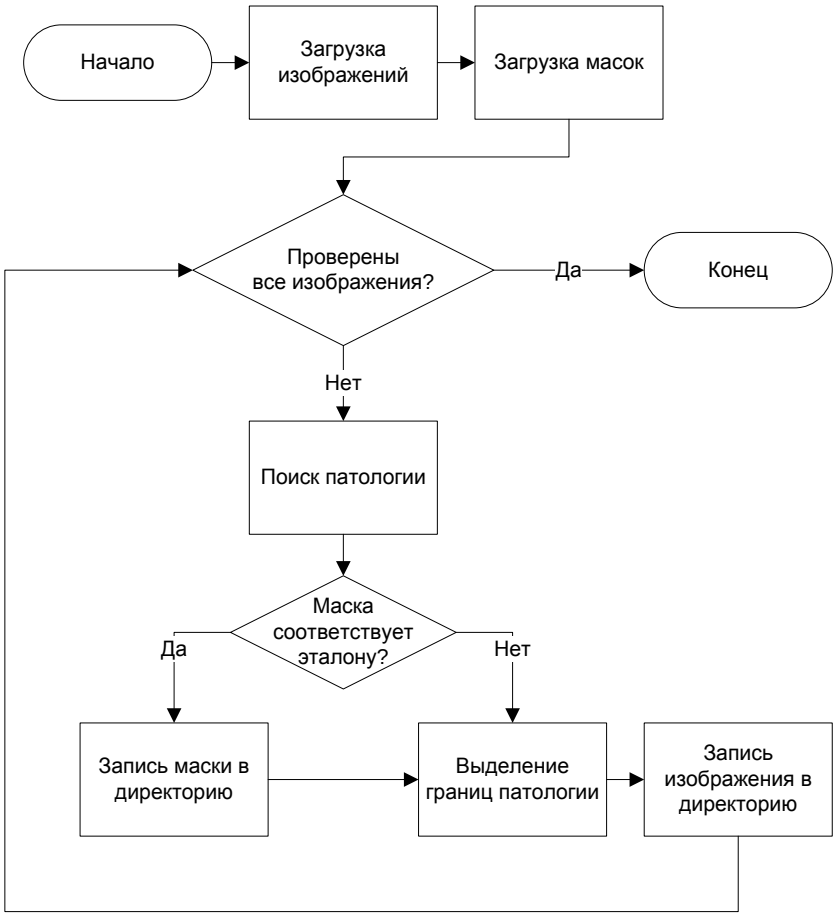

Рисунок 2.14 – Блок-схема алгоритма поиска патологий

Загрузка масок с патологиями реализована аналогично загрузке изображений из директории, но для масок указывается директория «Masks».

После загрузки масок запускается цикл поиска патологии внутри флюорографического снимка. Поиск осуществляется с заданным отличием оттенков цвета между шаблоном и набором пикселей исходного изображения, при этом исключается поиск маски на фоне (фон изображения имеет 255 оттенок цвета). Допустимое отличие оттенков задается с помощью переменной shade. Например, если текущий оттенок цвета пикселя на маске с патологией равен 50, а значение переменной shade равно 10, удовлетворять условию будут все пиксели флюорографического изображения в диапазоне оттенков от 40 до 60 включительно.

Если текущий пиксель удовлетворяет условию, увеличивается счетчик прошедших проверку пикселей – countTrueMask. Если текущий пиксель не удовлетворяет условию, увеличивается счетчик не прошедших проверку пикселей – countFalseMask.

Код условия проверки на отличие оттенков цвета между шаблоном и набором пикселей исходного изображения приведен на рисунке 2.15.

```
shade = 10;
countTrueMask = 0;
countFalseMask = 0;

if (Array(i,j) ~= 255)
    if ((Array(i + x - 1,j + y - 1) >= (Mask(x,y) - shade))
        && (Array(i + x - 1,j + y - 1) <= (Mask(x,y) + shade)))
        countTrueMask = countTrueMask + 1;
    else
        countFalseMask = countFalseMask + 1;
    end
end
```

Рисунок 2.15 – Код условия проверки на отличие оттенков цвета

В переменной percentForTrueMask задается минимальный процент пикселей на флюорографическом снимке, удовлетворяющих условию проверки на отличие оттенков цвета между шаблоном и набором пикселей исходного изображения, которые требуются для нахождения патологии. Переменная

trueMask вычисляет количество таких пикселей для каждой маски на основе установленного значения в переменной percentForTrueMask.

Требуемое количество совпадающих с маской пикселей для нахождения патологии вычисляется по формуле:

$$N = n * m * p, \tag{2.3}$$

где N – количество пикселей, пикс;

n – ширина маски, пикс;

m – высота маски, пикс;

p – процент совпадающих с маской пикселей.

Если счетчик прошедших проверку пикселей в переменной countTrueMask достигает значения переменной trueMask – совпадение по маске считается установленным. Значения цветов всех пикселей на флюорографическом снимке, которые находятся в области найденной маски, записываются в массив Mask для последующего создания нового шаблона маски с патологией.

Код создания массива с новым шаблоном маски с патологией приведен на рисунке 2.16.

```
percentForTrueMask = 0.9;
[rowMask,columnMask] = size(Mask);
trueMask = round(rowMask * columnMask * percentForTrueMask);

if (countTrueMask == trueMask)
    Mask = imread(['.\Masks\' Masks{1,numberMask}]);
    for row = 1 : rowMask
        for column = 1 : columnMask
            Mask(row,column) = Array(i + row - 1,j + column - 1);
        end
    end
end
```

Рисунок 2.16 – Код создания массива с новым шаблоном маски с патологией

Прежде чем добавить новый шаблон маски с патологией к другим шаблонам, необходимо проверить, может ли найденная маска с патологией считаться эталоном.

Значения оттенков цвета патологии увеличиваются от ее центра к границам. Самые низкие значения цвета в центре патологий находятся в диапазоне от 25 до 40 оттенка, а самые высокие значения цвета у границ патологий находятся в диапазоне от 60 до 70 оттенка.

Область патологии относительно размерности шаблона маски с патологией вычисляется по формуле:

$$k = \frac{\pi r^2}{n * m}, \tag{2.4}$$

где k – отношение размера патологии к размерности маски;

π – число Пи;

r – радиус патологии, пикс;

n – ширина маски, пикс;

m – высота маски, пикс.

Область патологии от центра до ее границ занимает чуть больше половины размерности маски. Отношение размера патологии к размерности маски на основе формулы (2.4) представлено в таблице 2.3.

Таблица 2.3 – Отношение размера патологии к размерности маски

Маска	Ширина, пикс	Высота, пикс	Радиус патологии, пикс	Отношение
Mask_1.jpg	32	32	13	0.51849
Mask_2.jpg	34	34	14	0.53266
Mask_3.jpg	36	36	15	0.54542
Mask_4.jpg	38	38	16	0.55696
Mask_5.jpg	40	40	16	0.50265

Гистограмма шаблонов масок с патологиями представлена на рисунке 2.17. На данной гистограмме видно, что количество пикселей с оттенками цвета ниже 70 занимают чуть больше половины размерности маски.

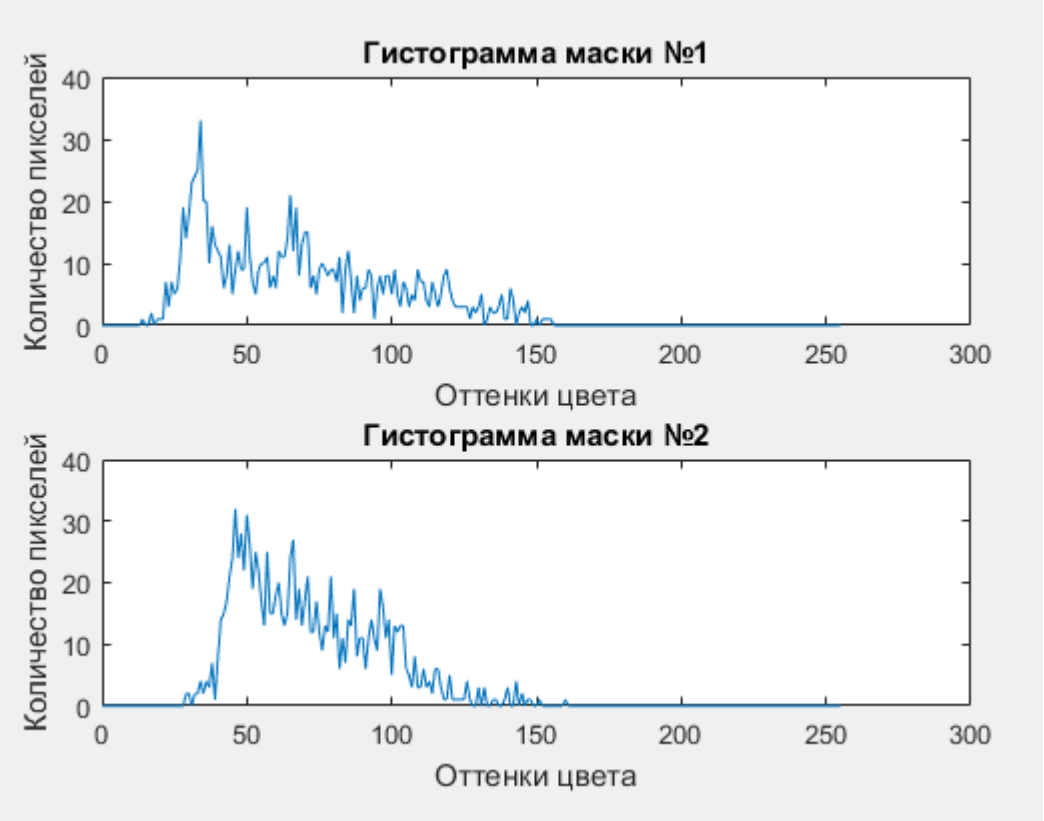

Рисунок 2.17 – Гистограмма шаблонов масок с патологиями

Следовательно, эталоном можно считать маску, половину размерности которой занимают пиксели со значениями цвета ниже 70 оттенка.

Для подсчета количества пикселей со значением цвета ниже 70 оттенка введена переменная sumShade. Если количество таких пикселей в маске больше половины ее размерности, новый шаблон маски с патологией добавляется к другим патологиям в директорию «Masks». Переменная cointMask содержит значение общего числа масок, а каждая маска имеет свой номер в порядке нумерации, поэтому при записи новой маски к ее имени добавляется следующий номер, а поиск патологий проводится с учетом добавленной маски.

Код записи нового шаблона маски с патологией в директорию «Masks» приведен на рисунке 2.18.

```
if (countTrueMask == trueMask)
    sumShade = 0;
    for row = 1 : rowMask
        for column = 1 : columnMask
            if (Mask(row,column) < 70)
                sumShade = sumShade + 1;
            end
        end
    end
    if (sumShade > round(rowMask * columnMask / 2))
        countMask = countMask + 1;
        imwrite(Mask, ['.\Masks\Mask_'
            int2str(countMask) '.' extension], extension);
    end
end
```

Рисунок 2.18 – Код записи нового шаблона маски с патологией в директорию «Masks»

Если на флюорографическом снимке установлено совпадение по маске, на исходном изображении производится выделение границ совпадающей области. Границей области является белый (255 оттенок цвета) квадрат с толщиной 2 пикселя. Размер данного квадрата на 10 пикселей больше совпадающей области для каждой из его сторон. Затем производится запись изображения с выделенными границами патологии в директорию «Pathologies». К исходному имени файла добавляется символ решетки «#», которым будут отмечаться все изображения с найденными патологиями.

Код выделения границ совпадающей с маской области приведен на рисунке 2.19.

```
if (countTrueMask == trueMask)
    for row = 1 - 10 : rowMask + 10
        for column = 1 - 10 : columnMask + 10
            Array(i - 10,j + column - 1) = 255;
            Array(i + 10 + rowMask - 1,j + column - 1) = 255;
            Array(i + row - 1,j - 10) = 255;
            Array(i + row - 1,j + 10 + columnMask - 1) = 255;
            Array(i - 9,j + column - 1) = 255;
            Array(i + 9 + rowMask - 1,j + column - 1) = 255;
            Array(i + row - 1,j - 9) = 255;
            Array(i + row - 1,j + 9 + columnMask - 1) = 255;
        end
    end
    imwrite(Array, ['.\Pathologies\#'
        Images{1,countImages}], extension);
end
```

Рисунок 2.19 – Код выделения границ совпадающей с маской области

Код алгоритма поиска патологий на флюорографических снимках приведен в приложении Б.

2.3 Реализация алгоритма проверки масок

Прежде чем применить алгоритм поиска патологий на флюорографическом снимке, необходимо убедиться, что все маски в директории «Masks» являются эталонами.

Блок-схема алгоритма проверки масок представлена на рисунке 2.20.

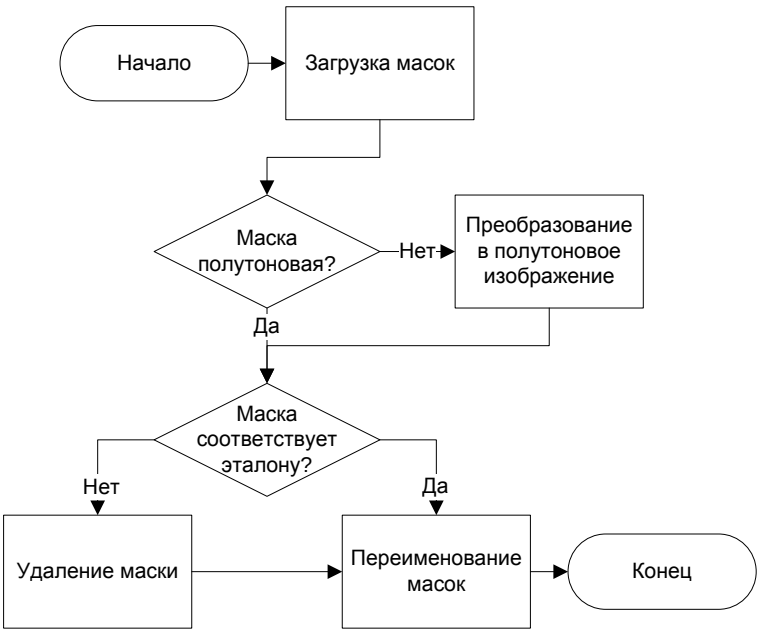

Рисунок 2.20 – Блок-схема алгоритма проверки масок

Маска считается эталоном, если она является полутоновым изображением (256 оттенков цвета). Так как любое полутоновое изображение имеет 256 оттенков цвета, глубина цвета маски должна составлять 8 бит. Функция imfinfo записывает информацию об изображении и способе его хранения в переменную info. Затем производится проверка глубины цвета маски. Если она не равна 8 бит, происходит преобразование маски в полутоновое изображение с помощью функции rgb2gray. Функция rgb2gray создает полутоновое изображение, преобразуя RGB-составляющие пикселей полноцветного изображения в соответствующие им значение яркости. После преобразования маски в полутоновое изображение она записывается в директорию «Masks», имя маски не меняется.

Код проверки маски и ее преобразования в полутоновое изображение приведен на рисунке 2.21.

```
extension = ('jpg');

directory = dir(['.\Masks\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[m,n] = size(Images);
for numberMask = 1 : n
    info = imfinfo(['.\Masks\' Images{1,numberMask}]);
    if (info.BitDepth ~= 8)
        gs = rgb2gray(imread(['.\Masks\' Images{1,numberMask}]));
        imwrite(gs, ['.\Masks\' Images{1,numberMask}], extension);
    end
end
```

Рисунок 2.21 – Код проверки маски и ее преобразования в полутоновое изображение

Маска считается эталоном, если половину ее размерности занимают пиксели со значениями цвета ниже 70 оттенка. Проверка оттенков цвета пикселя в маске аналогична алгоритму записи нового шаблона маски с патологией в директорию «Masks», который описан в подразделе 2.2, а код приведен на рисунке 2.3. Если проверяемая маска не соответствует требованиям эталона, она удаляется.

Для корректной работы алгоритмов программы, использующих загрузку шаблонов масок с патологиями из директории «Masks», имена масок должны быть заданы в порядке нумерации. Для этого разработан алгоритм переименования масок в нужном порядке. Чтобы исключить ошибки при простом добавлении цифр к маскам в порядке нумерации, алгоритм меняет имена шаблонов два раза. Сначала маскам присваиваются имена вида «Rename_%d.jpg», где %d является числом в порядке нумерации. Затем происходит аналогичный алгоритм, где изображениям присваиваются имена вида «Mask_%d.jpg», где %d также является числом в порядке нумерации.

Код переименования масок приведен на рисунке 2.22.

```
cd('.\Masks\');
dirData = dir(['*.' extension]);
fileNames = {dirData.name};
for iFile = 1:numel(fileNames)
    newName = sprintf(['Rename_%d.' extension], iFile);
    movefile(fileNames{iFile},newName);
end
cd('.\..');

cd('.\Masks\');
dirData = dir(['*.' extension]);
fileNames = {dirData.name};
for iFile = 1:numel(fileNames)
    newName = sprintf(['Mask_%d.' extension],iFile);
    movefile(fileNames{iFile},newName);
end
cd('.\..');
```

Рисунок 2.22 – Код переименования масок

Код алгоритма проверки масок приведен в приложении В.

Реализованы алгоритмы выделения границ на рисунке, поиска патологий на флюорографическом снимке и проверки масок, а также описана их работа.

3 ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Соответствие разработанного программного обеспечения формализованным требованиям

Разработанное программное обеспечение для выделения границ легочного (сосудистого) рисунка содержит алгоритм загрузки в программу массива цветов полутонового (черно-белого) изображения – флюорографический снимок. К загруженному изображению применяется алгоритм выделения границ легочного рисунка на основе алгоритма бинаризации изображения. Для проверки исходного изображения на наличие патологий загружаются массивы цветов шаблонов с патологиями (маски) аналогично алгоритму загрузки массива цветов полутонового изображения, а затем применяется алгоритм поиска патологии внутри выделенной области на флюорографическом снимке по принципу поиска массива внутри массива.

Поиск патологии осуществляется с заданной в алгоритме программы погрешностью – отличием цветов между шаблоном маски с патологией и набором пикселей исходного изображения. Также внутри программы устанавливается допустимый процент не совпадающих с маской элементов.

При обнаружении сходства между маской и набором пикселей исходного изображения программа проверяет, может ли найденная маска с патологией считаться эталоном. Если маска с патологией проходит проверку, создается новый шаблон маски с патологией на основе совпадающих с маской элементов и добавляется в директорию к уже имеющимся шаблонам. Найденная область патологии на флюорографическом снимке выделяется, а затем изображение с выделенными границами патологии записывается в директорию с патологиями.

Алгоритм поиска патологий будет применяться к загруженному флюорографическому снимку до тех пор, пока не будет произведена проверка с применением всех имеющихся шаблонов масок с патологиями. Исходное изображение будет считаться нормой, если программа не обнаружит патологий на основе всех имеющихся шаблонов.

3.2 Тестирование работы алгоритмов программного обеспечения

Для удобства тестирования работы алгоритмов программного обеспечения для выделения границ легочного (сосудистого) рисунка был разработан простой интерфейс пользователя, содержащий окно для вывода информации и четыре кнопки: «проверка масок», «выделение легочного рисунка», «поиск патологий», «поиск патологий на области».

Кнопка «проверка масок» запускает алгоритм проверки шаблонов масок с патологиями, который определяет, может ли существующий в директории «Masks» шаблон маски с патологией считаться эталоном.

Действия алгоритма проверки масок отображаются в окне пользовательского интерфейса и представлены на рисунке 3.1.

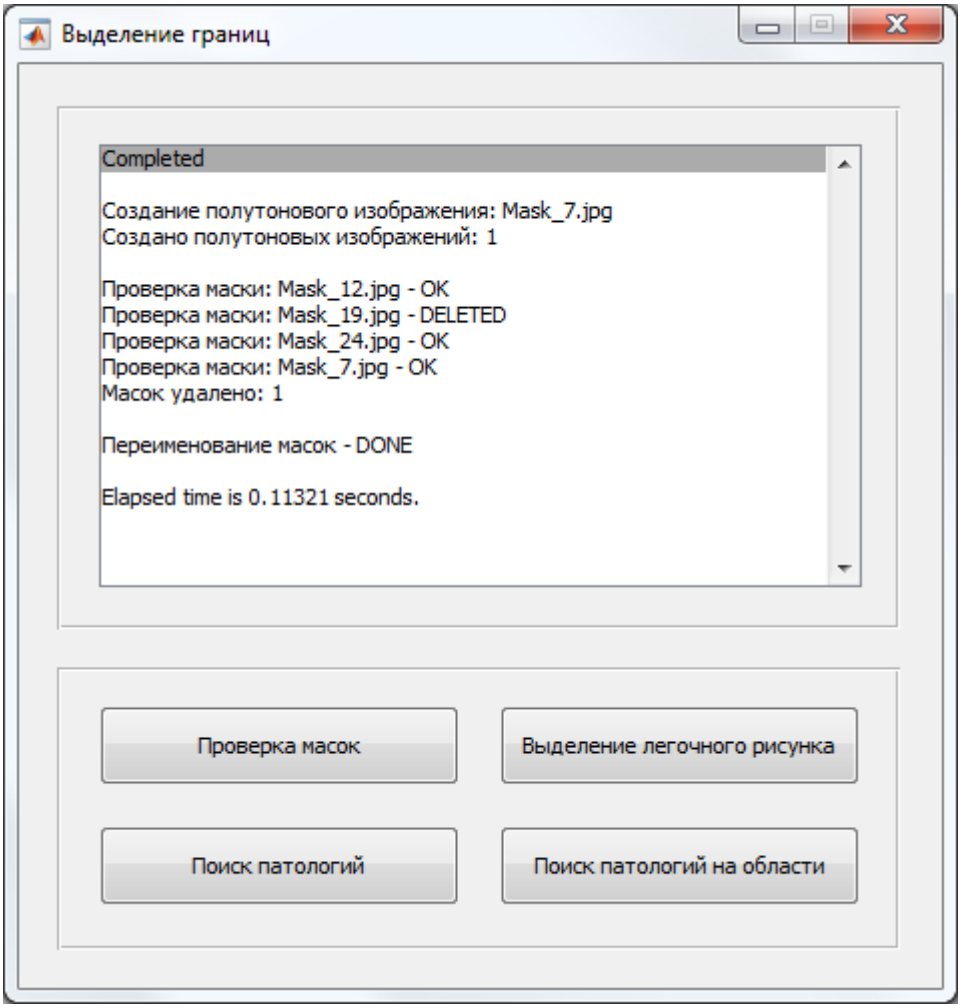

Рисунок 3.1 – Действия алгоритма проверки масок

На рисунке 3.2 представлены имена файлов в директории «Masks» до применения алгоритма проверки масок. На рисунке 3.3 представлены имена файлов в директории «Masks» после применения алгоритма проверки масок.

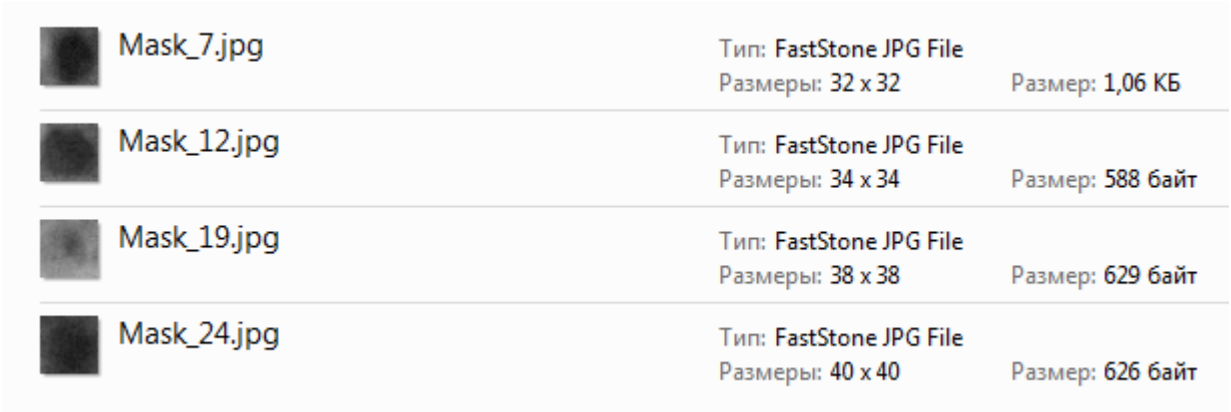

Рисунок 3.2 – Имена файлов в директории «Masks» до применения алгоритма проверки масок

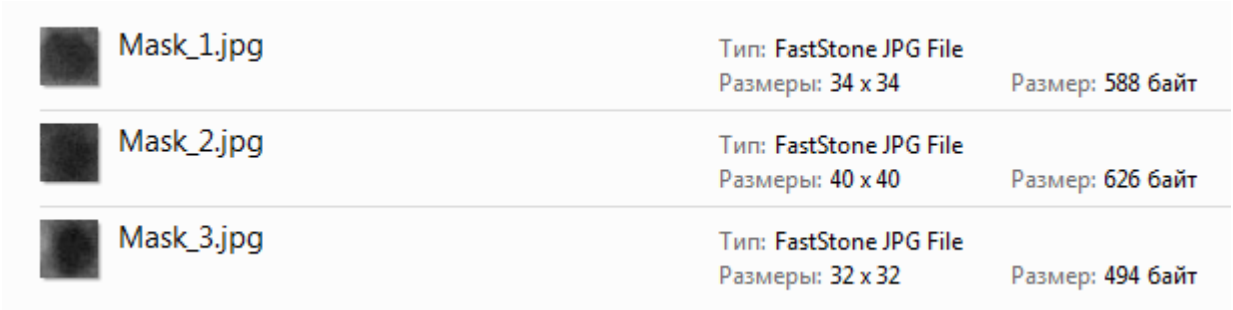

Рисунок 3.3 – Имена файлов в директории «Masks» после применения алгоритма проверки масок

Шаблон «Mask_7» не является полутоновым изображением, поэтому данная маска была преобразована в полутоновое изображение.

Шаблон «Mask_19» не может считаться эталоном, так как более половины размерности данной маски занимают пиксели со значениями цвета выше 70 оттенка, поэтому данный шаблон был удален из директории «Mask».

Так как шаблонам масок с патологиями заданы имена в хаотичном порядке, произведено переименование масок.

Кнопка «выделение легочного рисунка» запускает алгоритм выделения границ легочного рисунка, который загружает все изображения из директории

«Images». К каждому загруженному изображению применяется алгоритм выделения легочной области на основе алгоритма бинаризации изображения, затем полученные в результате работы алгоритма изображения записываются в директорию «Borders».

Действия алгоритма выделения границ легочного рисунка отображаются в окне пользовательского интерфейса и представлены на рисунке 3.4.

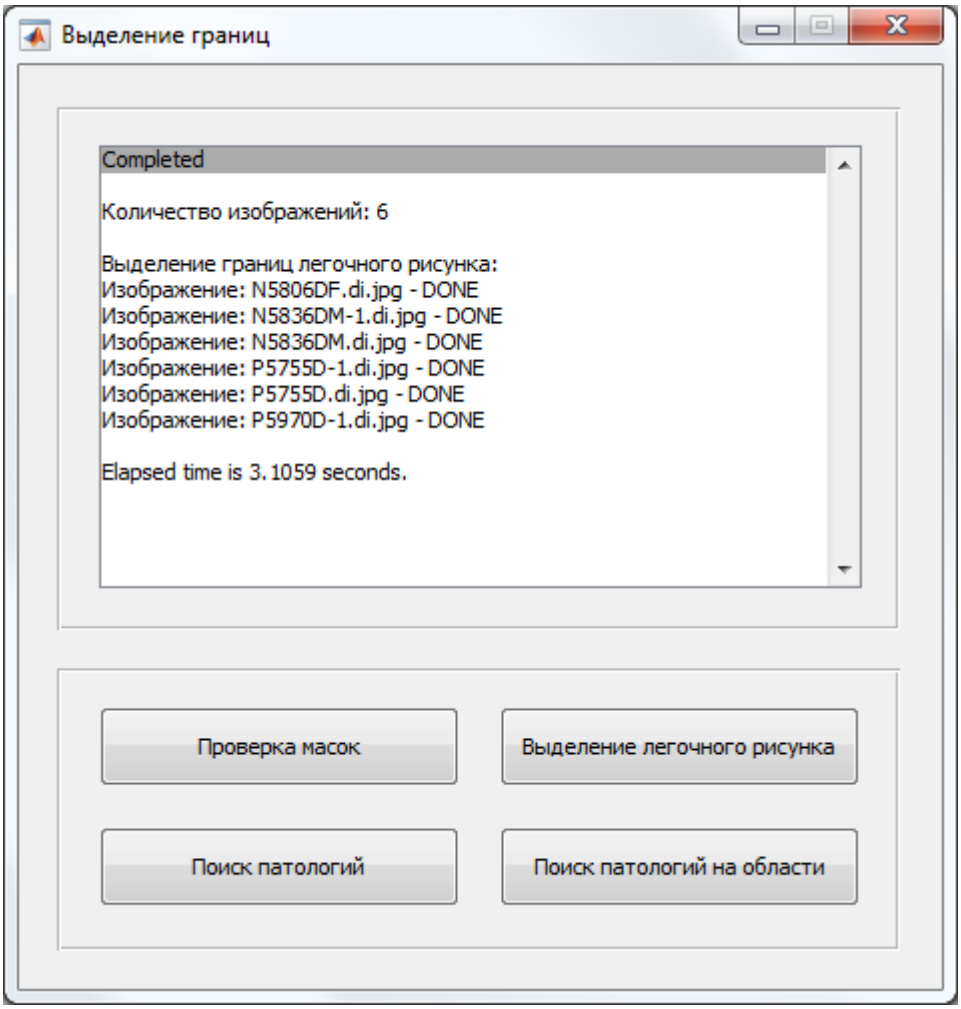

Рисунок 3.4 – Действия алгоритма выделения границ легочного рисунка

Записанные в директорию «Borders» в результате работы алгоритма выделения границ легочного рисунка изображения представлены на рисунке 3.5.

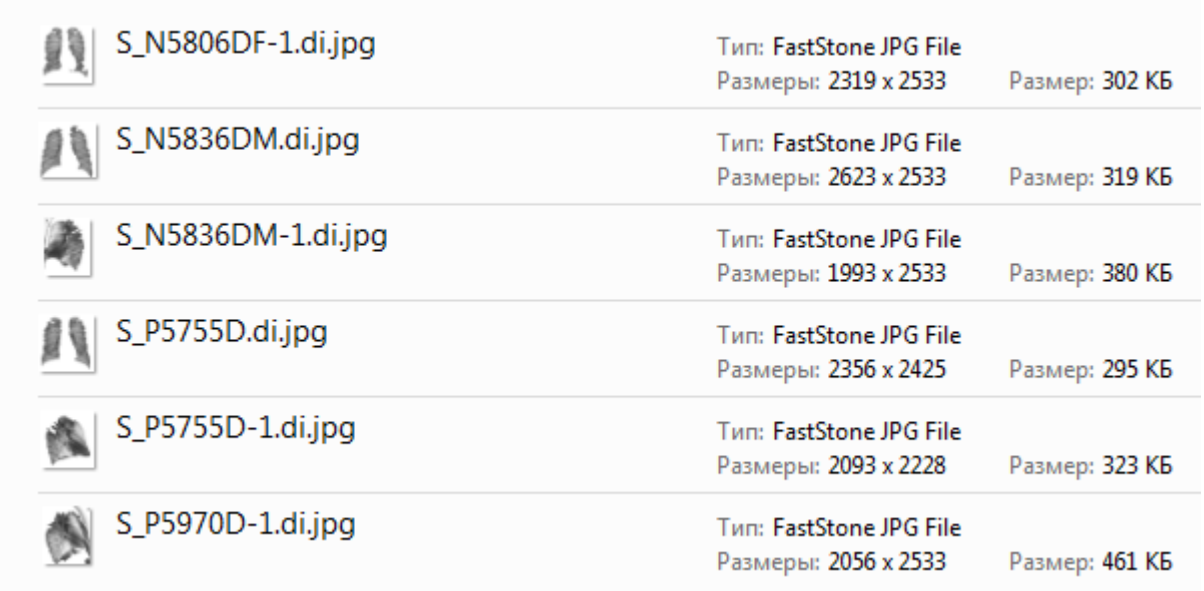

Рисунок 3.5 – Записанные в директорию «Borders» изображения

Кнопка «поиск патологий» запускает алгоритм поиска патологий на исходных флюорографических снимках, которые загружаются из директории «Images». Для проверки исходных изображений на наличие патологий сначала загружаются шаблоны масок с патологиями из директории «Masks», а затем применяется алгоритм поиска патологий на флюорографическом снимке. Полученные в результате работы алгоритма изображения записываются в директорию «Pathologies» с добавлением к имени файла символа решетки «#».

Действия алгоритма поиска патологий на исходных флюорографических снимках отображаются в окне пользовательского интерфейса и представлены на рисунке 3.6.

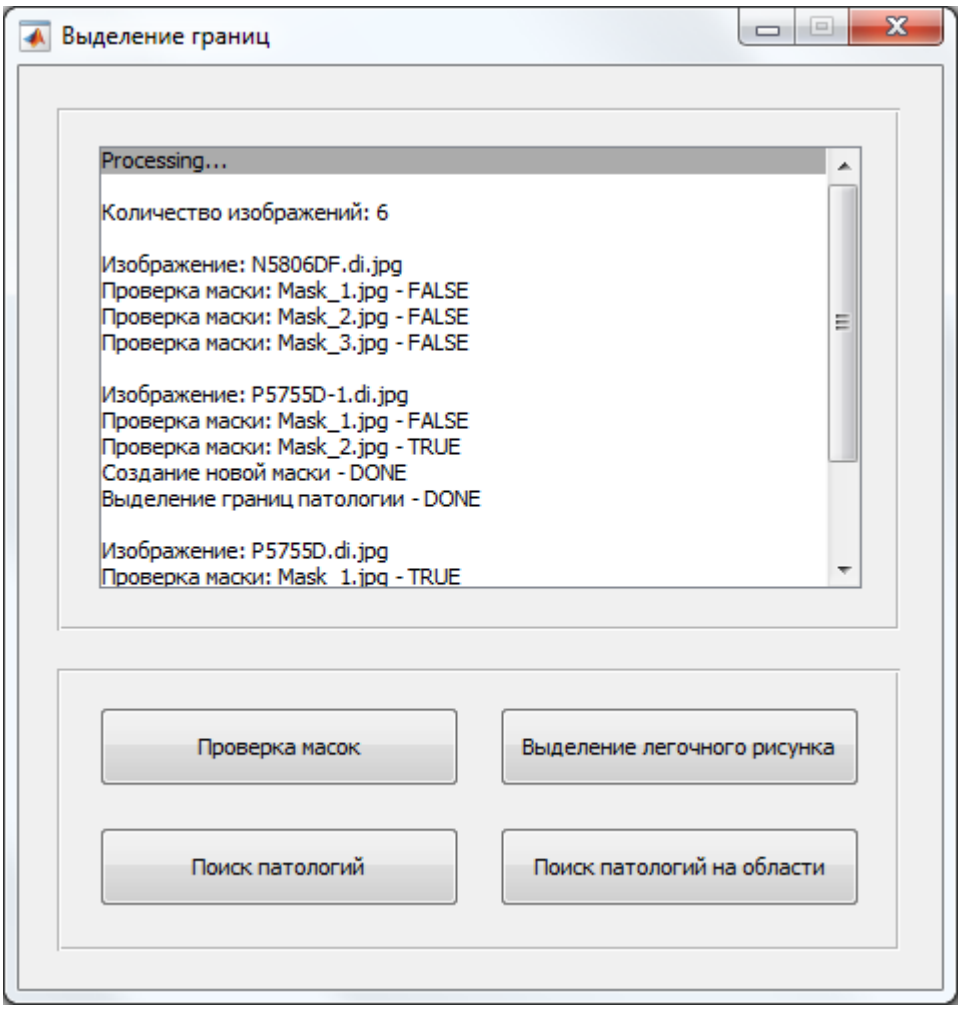

Рисунок 3.6 – Действия алгоритма поиска патологий на исходных флюорографических снимках

Записанные в директорию «Pathologies» в результате работы алгоритма поиска патологий на исходных флюорографических снимках изображения представлены на рисунке 3.7.

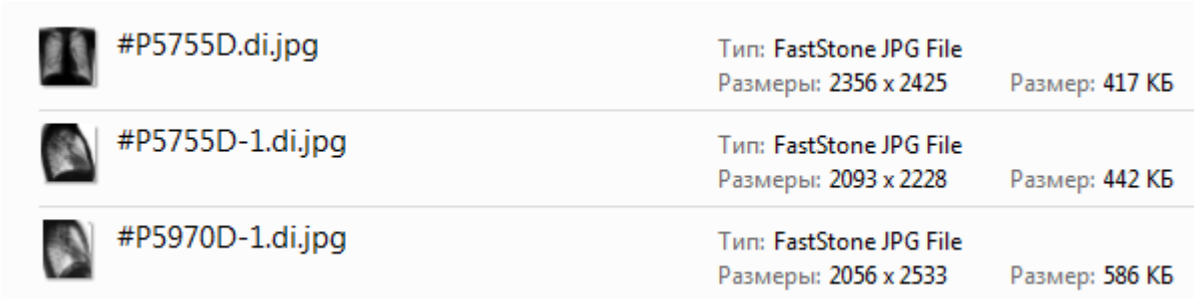

Рисунок 3.7 – Записанные в директорию «Pathologies» изображения

Также в результате работы алгоритма поиска патологий на исходных флюорографических снимках были созданы новые шаблоны масок с патологиями и добавлены к другим шаблонам в директорию «Masks».

Добавленные в директорию «Masks» в результате работы алгоритма поиска патологий на исходных флюорографических снимках шаблоны масок с патологиями представлены на рисунке 3.8.

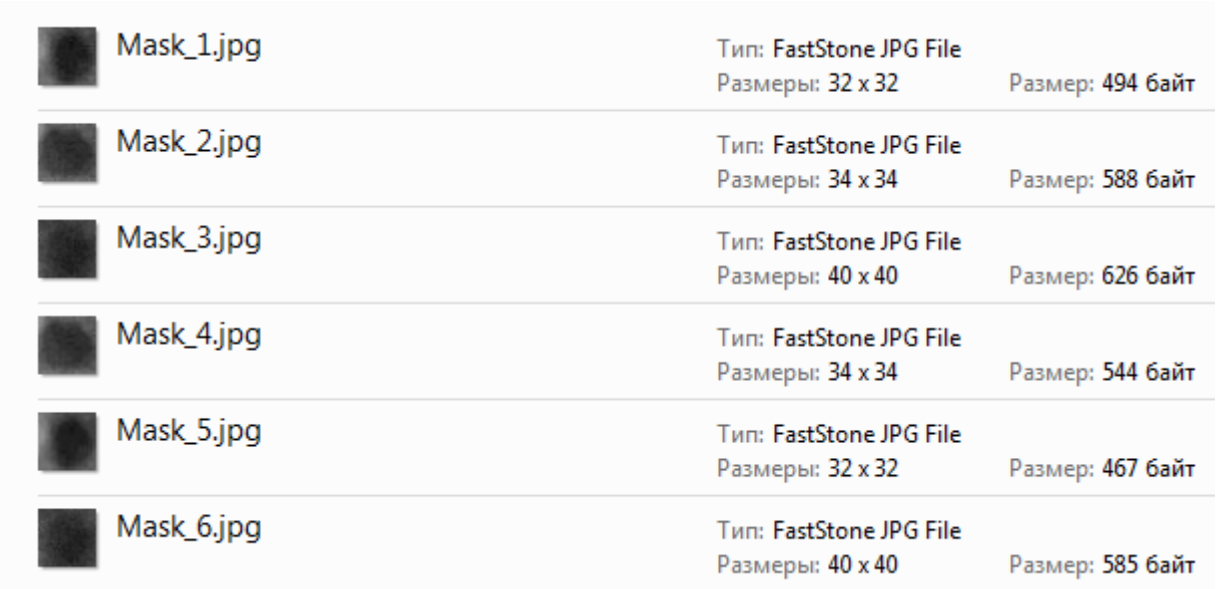

Рисунок 3.8 – Добавленные в директорию «Masks» шаблоны масок с патологиями

Кнопка «поиск патологий на области» запускает основанный на алгоритме бинаризации изображения алгоритм выделения границ легочного рисунка на флюорографических снимках, которые загружаются из директории «Images». Для проверки изображений с выделенными границами легочного рисунка на наличие патологий сначала загружаются шаблоны масок с патологиями из директории «Masks», а затем применяется алгоритм поиска патологий. Полученные в результате работы алгоритма изображения записываются в директорию «Pathologies» с добавлением к имени файла символов «S#».

Как и в случае с алгоритмом поиска патологий на исходных флюорографических снимках в результате работы алгоритма поиска патологий на изображениях с выделенными границами легочного рисунка были созданы

новые шаблоны масок с патологиями и добавлены к другим шаблонам в директорию «Masks».

Действия алгоритма поиска патологий на изображениях с выделенными границами легочного рисунка отображаются в окне пользовательского интерфейса и представлены на рисунке 3.9.

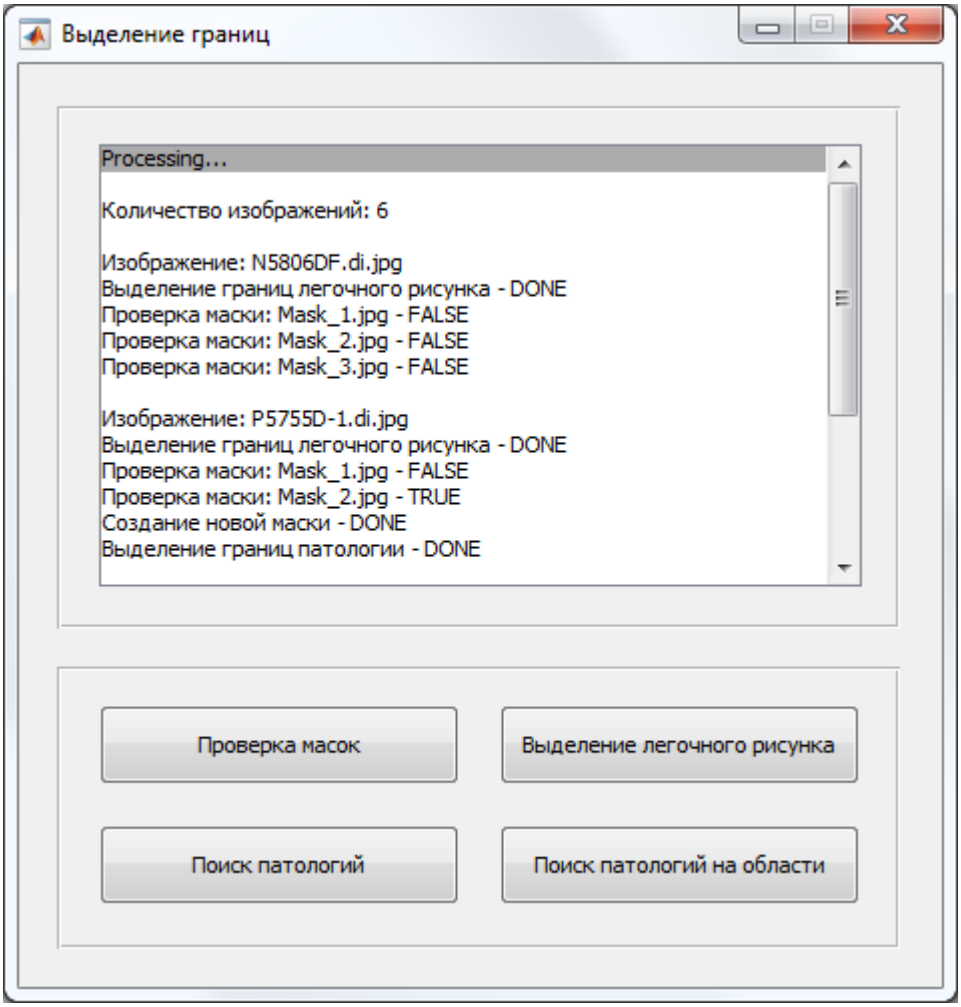

Рисунок 3.9 – Действия алгоритма поиска патологий на изображениях с выделенными границами легочного рисунка

Записанные в директорию «Pathologies» в результате работы алгоритма поиска патологий на изображениях с выделенными границами легочного рисунка изображения представлены на рисунке 3.10.

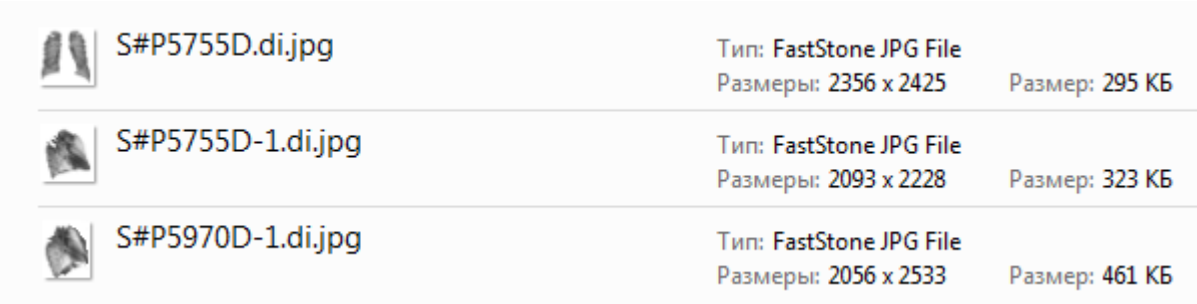

Рисунок 3.10 – Записанные в директорию «Pathologies» изображения

Разработанные алгоритмы соответствуют формализованным требованиям. Произведено тестирование алгоритмов программного обеспечения с помощью интерфейса пользователя, некорректное поведение программы не было обнаружено.

ЗАКЛЮЧЕНИЕ

В ходе выпускной квалификационной работы была описана актуальность рассматриваемой темы, определены объект и предмет выпускной квалификационной, поставлена цель и выявлены задачи. Также был проведен анализ состояния вопроса, а именно – проведен обзор существующих программных аналогов. Кроме того, были формализованы требования к разрабатываемому программному продукту, описан язык программирования, с помощью которого разработаны алгоритмы программы, а также была описана работа данного программного обеспечения.

В результате разработки программного обеспечения были созданы все алгоритмы, описанные в формализации требований к разрабатываемому программному продукту, а работа алгоритмов программы полностью соответствует указанным требованиям. Также была протестирована работа программного обеспечения с помощью пользовательского интерфейса на наличие сбоев и ошибок, некорректное поведение работы алгоритмов не было обнаружено.

Разработанное в ходе выпускной квалификационной работы программное обеспечение является самообучаемым, однако имеет низкую скорость работы алгоритма поиска патологий за счет большого количества шаблонов масок с патологиями, что компенсируется высокой точностью их поиска на изображении, поэтому программный продукт имеет перспективу улучшения и развития. Данное программное обеспечение может быть дополнено новыми функциональными алгоритмами, работа существующих алгоритмов может быть распределена между несколькими потоками для увеличения производительности программы, а погрешность поиска патологий может быть задана в окне пользовательского интерфейса.

Апробация данной работы была представлена в научно-практической конференции «Студенческие Дни науки в ТГУ – 2017», которая проводилась в

Тольяттинском государственном университете с 3 по 15 апреля 2017 года на кафедрах институтов в виде семинаров.

Кроме того, апробация работы была представлена и опубликована на III научно-практической всероссийской конференции (школе-семинаре) молодых ученых «Современные исследования в области естественных и технических наук», которая проводилась в Тольяттинском государственном университете с 24 по 25 апреля 2017 года в заочной форме.

Также апробация работы была представлена и опубликована на XIV Всероссийской студенческой конференции «Информационные технологии в современном мире – 2017», которая проводилась организаторами АНО ВО «Гуманитарный университет» 5 мая 2017 года в очно-заочной форме.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Нормативно-правовые акты

1. ГОСТ 2.105 – 95. Общие требования к текстовым документам [Текст]. – М.: Изд-во стандартов, 1996. – 29 с. – (Единая система конструкторской документации).

2. ГОСТ 7.1-2003. Библиографическая запись. Библиографическое описание документа.

3. ГОСТ 7.32-2001. Отчет о научно-исследовательской работе. Структура и правила оформления.

4. ГОСТ 7.82-2001. Библиографическая запись. Библиографическое описание электронных ресурсов.

5. ГОСТ 19.701 – 90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения (ИСО 5807–85) [Текст]. Введен 1992–01–01. – М.: Изд-во стандартов, 1992. – 14 с. – (Единая система программной документации).

Научная и методическая литература

6. Алексеев Е.Р., Чеснокова О.В. MATLAB 7. Самоучитель. Издательство: «НТ Пресс», 2006 г. – 464 стр.

7. Браженко Н.А. Туберкулез органов дыхания. Издательство: СпецЛит, 2012 г. – 368 стр.

8. В. Дьяконов, И. Абраменкова. MATLAB. Обработка сигналов и изображений. Специальный справочник. Издательство: Питер, 2002 г. – 608 стр.

9. В.П. Дьяконов. MATLAB 6.0/6.1/6.5/6.5 + SP1 + Simulink 4/5. Обработка сигналов и изображений. Москва: СОЛОН-Пресс, 2004 г. – 592 стр.

10. В.П. Дьяконов. Matlab 6.5 SP1/7 + Simulink 5/6. Работа с изображениями и видеопотоками. Москва: СОЛОН-Пресс, 2005 г. – 400 стр.

11. Егоров, А.Г. Правила оформления выпускных квалификационных работ по программам подготовки бакалавра и специалиста: учебно-

методическое пособие / А.Г. Егоров, В.Г. Виткалов, Г.Н. Уполовникова, И.А. Живоглядова – Тольятти: ТГУ, 2012. – 135 с.

12. Зимина В.Н., Кошечкин В.А., Кравченко А.В. Туберкулез и ВИЧ-инфекция у взрослых. Руководство. Издательство: ГЭОТАР-Медиа, 2014 г. – 224 стр.

13. Курбатова Е.А. MATLAB 7. Самоучитель. Издательство: Вильямс, 2005 г. – 256 стр.

14. Максимова Е.И. Алгоритм обнаружения образований на снимках компьютерного томографа с использованием кластеризации / Е.И. Максимова ; науч. рук. П.А. Хаустов // Современные техника и технологии : сборник трудов XXI международной научной конференции студентов, аспирантов и молодых ученых, Томск, 5-9 октября 2015 г. : в 2 т. – Томск : Изд-во ТПУ, 2015. – Т. 2. – [С. 39-42].

15. Матиас Хофер. Рентгенологическое исследование грудной клетки. Практическое руководство. Издательство: «Медицинская литература», 2008 г. 224 стр.

16. Положение о выпускной квалификационной работе. – Тольятти: ТГУ. – 2014.

17. Поршнев С.В. MATLAB 7. Основы работы и программирования. Учебник. Издательство: «Бином. Лаборатория знаний», 2006 г. – 320 стр.

18. П. Рудаков, В. Сафонов. Обработка сигналов и изображений Matlab 5.x. Издательство: Диалог-МИФИ, 2000 г. – 416 стр.

19. Р. Гонсалес, Р. Вудс, С. Эддинс. Цифровая обработка изображений в среде MATLAB. Москва: Техносфера, 2006 г. – 616 стр.

20. Федеральный государственный образовательный стандарт высшего образования направления подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем (уровень бакалавриата), утвержденный приказом Минобрнауки России от 12.03.2015 N222.

21. Хофер Матиас. Компьютерная томография. Базовое руководство. Издательство: Медицинская литература, 2011 г. – 232 с.

Электронные ресурсы

22. Всемирная организация здравоохранения. Туберкулез [Электронный ресурс]. – Электрон. дан. – [2017]. – Режим доступа : http://www.who.int/mediacentre/factsheets/fs104/ru.

23. Медицинская энциклопедия. Рентгенологическое исследование [Электронный ресурс]. – Электрон. дан. – [2017]. – Режим доступа : http://www.medical-enc.ru/16/rentgenologicheskoe_issledovanie.shtml.

24. Медицинская энциклопедия. Флюорография [Электронный ресурс]. – Электрон. дан. – [2017]. – Режим доступа : http://www.medical-enc.ru/20/fluorography.shtml.

25. MathWorks. MATLAB [Электронный ресурс]. – Электрон. дан. – [2017]. – Режим доступа : http://matlab.ru/products/matlab.

Литература на иностранном языке

26. Gérard Blanchet. Digital Signal and Image Processing using MATLAB, Volume 2: Advances and Applications: The Deterministic Case, 2e. John Wiley & Sons, Inc., 2015.

27. Gérard Blanchet. Digital Signal and Image Processing Using MATLAB, Volume 3: Advances and Applications, The Stochastic Case, 2nd Edition. John Wiley & Sons, Inc., 2015.

28. Maurice Charbit, Gérard Blanchet. Digital Signal and Image Processing using MATLAB. ISTE Ltd, 2006.

29. Maurice Charbit, Gérard Blanchet. Digital Signal and Image Processing using MATLAB, Volume 1. ISTE Ltd, 2015

30. Tamalika Chaira. Medical Image Processing: Advanced Fuzzy Set Theoretic Techniques. CRC Press, Inc., 2015.

ПРИЛОЖЕНИЕ А

Код алгоритма выделения границ легочного рисунка

```
extension = ('jpg');
VDel = 0.125;

directory = dir(['.\Images\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[mImages,nImages] = size(Images);
for countImages = 1 : nImages
    Array = imread(['.\Images\' Images{1,countImages}]);

    white = 0;
    [rowArray,columnArray] = size(Array);
    limit = round(rowArray * columnArray * 0.001);
    for i = 1 : rowArray
        if (white > limit)
            break
        end
        for j = 1 : columnArray
            if (white > limit)
                break
            end
            if (Array(i,j) > 240)
                white = white + 1;
            end
        end
    end

    if (white > limit)
        gamma=imadjust(Array, [0 100]/255);
    else
        gamma=imadjust(Array, [0 200]/255);
    end

    VObj = round(rowArray * columnArray * VDel);
    RibCage = im2bw(gamma, graythresh(gamma));
    RibCage = bwareaopen(RibCage, VObj);
    SE = strel('disk', 5);
    RibCage = imclose(RibCage, SE);
    RibCage=imfill(RibCage,'holes');

    [rowArray,columnArray] = size(Array);
    for i = 1 : rowArray
        for j = 1 : columnArray
            if (RibCage(i,j) == 0)
                Array(i,j) = 255;
            end
        end
    end

    imwrite(Array, ['.\Borders\S_' Images{1,countImages}], extension);
end
```

ПРИЛОЖЕНИЕ Б

Код алгоритма поиска патологий на флюорографических снимках

```
extension = ('jpg');
shade = 10;
percentForTrueMask = 0.9;
countCreating = 0;

directory = dir(['.\Images\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[mImages,nImages] = size(Images);
for countImages = 1 : nImages
    Array = imread(['.\Images\' Images{1,countImages}]);
    [rowArray,columnArray] = size(Array);
    found = false;
    directory = dir(['.\Masks\*.' extension]);
    Masks = {directory(~[directory.isdir]).name};
    [mMasks,nMasks] = size(Masks);
    countMask = nMasks;
    for numberMask = 1 : countMask
        if (found == true)
            break
        end
        Mask = imread(['.\Masks\' Masks{1,numberMask}]);
        [rowMask,columnMask] = size(Mask);
        trueMask = round(rowMask * columnMask * percentForTrueMask);
        falseMask = round(rowMask * columnMask *
            (1 - percentForTrueMask));
        for i = 1 : rowArray - rowMask + 1
            if (found == true)
                break
            end
            for j = 1 : columnArray - columnMask + 1
                if (found == true)
                    break
                end
                countTrueMask = 0;
                countFalseMask = 0;
                if (Array(i,j) ~= 255)
                    for x = 1 : rowMask
                        if (countFalseMask == falseMask)
                            break
                        end
                        for y = 1 : columnMask
                            if (countFalseMask == falseMask)
                                break
                            end
                            if ((Array(i + x - 1,j + y - 1) >=
                                (Mask(x,y) - shade)) && (Array(i + x - 1,
                                    j + y - 1) <= (Mask(x,y) + shade)))
                                countTrueMask = countTrueMask + 1;
                                if (countTrueMask == trueMask)
                                    found = true;
                                    Mask = imread(['.\Masks\' Masks{1,
                                    numberMask}]);
                                    for row = 1 : rowMask
                                        for column = 1 : columnMask
                                            Mask(row,column) = Array(i +
```

```
                                        row - 1,j + column - 1);
                                    end
                                end
                                sumShade = 0;
                                for row = 1 : rowMask
                                    for column = 1 : columnMask
                                        if (Mask(row,column) < 70)
                                            sumShade = sumShade + 1;
                                        end
                                    end
                                end
                                if (sumShade > round(rowMask *
                                    columnMask / 2))
                                    countMask = countMask + 1;
                                    imwrite(Mask, ['.\Masks\Mask_'
                                        int2str(countMask) '.'
                                        extension], extension);
                                    countCreating = countCreating + 1;
                                end
                                for row = 1 - 10 : rowMask + 10
                                    for column = 1 - 10 :
                                        columnMask + 10
                                        Array(i - 10,
                                            j + column - 1) = 255;
                                        Array(i + 10 + rowMask - 1,
                                            j + column - 1) = 255;
                                        Array(i + row - 1,
                                            j - 10) = 255;
                                        Array(i + row - 1,
                                            j + 10 + columnMask -
                                                1) = 255;
                                        Array(i - 9,
                                            j + column - 1) = 255;
                                        Array(i + 9 + rowMask - 1,
                                            j + column - 1) = 255;
                                        Array(i + row - 1,
                                            j - 9) = 255;
                                        Array(i + row - 1,
                                            j + 9 + columnMask -
                                                1) = 255;
                                    end
                                end
                                imwrite(Array, ['.\Pathologies\#'
                                    Images{1,countImages}],
                                        extension);
                                break
                            end
                        else
                            countFalseMask = countFalseMask + 1;
                        end
                    end
                end
            end
        end
    end
end
```

ПРИЛОЖЕНИЕ В

Код алгоритма проверки масок

```
extension = ('jpg');

grayscale = 0;
directory = dir(['.\Masks\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[m,n] = size(Images);
for numberMask = 1 : n
    info = imfinfo(['.\Masks\' Images{1,numberMask}]);
    if (info.BitDepth ~= 8)
        gs = rgb2gray(imread(['.\Masks\' Images{1,numberMask}]));
        imwrite(gs, ['.\Masks\' Images{1,numberMask}], extension);
        grayscale = grayscale + 1;
    end
end

countMask = 0;
directory = dir(['.\Masks\*.' extension]);
Images = {directory(~[directory.isdir]).name};
[m,n] = size(Images);
for numberMask = 1 : n
    Mask = imread(['.\Masks\' Images{1,numberMask}]);
    [rowMask,columnMask] = size(Mask);
    sumShade = 0;
    for i = 1 : rowMask
        for j = 1 : columnMask
            if (Mask(i,j) < 70)
                sumShade = sumShade + 1;
            end
        end
    end
    if (sumShade < round(rowMask * columnMask / 2))
        countMask = countMask + 1;
        delete(['.\Masks\' Images{1,numberMask}]);
    end
end

cd('.\Masks\');
dirData = dir(['*.' extension]);
fileNames = {dirData.name};
for iFile = 1:numel(fileNames)
        newName = sprintf(['Rename_%d.' extension], iFile);
        movefile(fileNames{iFile},newName);
end
cd('.\..');

cd('.\Masks\');
dirData = dir(['*.' extension]);
fileNames = {dirData.name};
for iFile = 1:numel(fileNames)
        newName = sprintf(['Mask_%d.' extension],iFile);
        movefile(fileNames{iFile},newName);
end
cd('.\..');
```