Санкт-Петербургский государственный университет
Факультет прикладной математики – процессов управления
Кафедра теории управления

Теплова Светлана Евгеньевна

Магистерская диссертация

Разработка программы визуализации медицинских данных средствами OpenGL

Направление 01.04.02 — «Прикладная математика и информатика»
Магистерская программа «Прикладная математика и информатика в задачах медицинской диагностики»

Научный руководитель,
кандидат физ.-мат. наук,
доцент Плоских В. А.

Санкт-Петербург
2017

Оглавление

ВВЕДЕНИЕ

В современном мире медицина является одной из важнейших отраслей в жизнедеятельности человека. С развитием технологий появляются все новые и новые способы диагностики и лечения различных заболеваний.

В последние годы на передний план в способах диагностики заболеваний выходит ядерная медицина. Ядерная медицина — это быстро развивающийся раздел медицины, в котором для лечения и диагностики различных заболеваний используются радионуклиды. Развиваются такие средства радиационной диагностики как ОФЭКТ [1], ПЭТ, различные томографические методы, не использующие радионуклиды, такие как КТ и МРТ.

Информацию, полученную от различных источников медицинской радиационной диагностики, необходимо визуализировать для возможности интерпретации её врачом.

Визуализация позволяет более эффективно извлекать информацию из данных, полученных во время диагностики. Компьютерная визуализация имеет гораздо большие преимущества в сравнении с другими аналогами, так как она позволяет обрабатывать полученные данные как непосредственно во время проведения исследования, так и проводить дальнейшую обработку после окончания исследования.

В настоящее время DICOM является одним из основных стандартов создания, хранения, передачи и визуализации медицинских изображений. Он позволяет хранить большое количество данных о пациенте, самом изображении, способе обследования и другие параметры исследования централизованно [2].

Работа с DICOM изображениями — это большая отрасль разработки в сфере медицинского программного обеспечения, позволяющая представить данные исследования в наиболее продуктивном для врача и пациента виде. Это

дает возможность визуализировать результаты обследования в том виде, который позволит врачу локализовать область исследования и определить, наиболее точно, диагноз пациента.

Разработка приложений для визуализации медицинских данных — важная область человеческой деятельности. С ростом числа способов и средств медицинской диагностики возникает необходимость в появлении соответствующих способов визуализации.

К сожалению, большинство современных приложений для визуализации медицинских данных разработаны иностранными компаниями, в основном, они являются проприетарным программным обеспечением. Поэтому необходимо разрабатывать собственные аналоги данного программного обеспечения на основе требований российских медиков.

ПОСТАНОВКА ЗАДАЧИ

В ходе данной работы было необходимо разработать приложение с поддержкой стандарта DICOM для визуализации изображений.

Данное приложение должно содержать в себе следующие основные функциональные возможности:

- Загрузка и визуализация DICOM изображений
- Возможность поиска и загрузки изображений с PACS-сервера.

И отвечать следующим требованиям:

- Кроссплатформенность
- Поддержка русскоязычного интерфейса
- Простота развертывания.

1. ТРЕБОВАНИЯ К ПО

Для более детального определения требований к программному продукту, сформулированных в предыдущем разделе, необходимо рассмотреть один из основных современных стандартов хранения и визуализации медицинских данных — DICOM. Важно понять спецификацию данного стандарта и определить области его использования. Еще одним необходимым шагом в формулировании требований к ПО является обзор существующих аналогов для определения их достоинств и недостатков.

Стандарт DICOM

DICOM – Digital Imaging and Communications in Medicine – это международный стандарт создания, хранения, передачи и визуализации медицинских изображений и связанной с ним информации о пациенте и проводимом обследовании. Стандарт DICOM представлен практически во всех отраслях радиологии, кардиологических исследованиях, и во многих диагностических устройствах таких как КТ, МРТ, ультразвуковые устройства, рентген и так далее. А также DICOM все чаще применяется и в других медицинских областях, например, в таких как офтальмология и стоматология [3, 5].

Среди десятков тысяч устройств для обработки изображений, DICOM является одним из самых широко распространенных стандартов медицинских изображений в мире. Буквально миллиарды DICOM изображений в данный момент используются в здравоохранении.

С момента первой публикации в 1993 году, DICOM ввел революционные изменения в практику радиологии, позволяющие заменить процесс получения рентгеновского снимка с использованием рентгеновской пленки на полностью цифровой процесс.

От использования в отделениях скорой помощи, до ЭКГ с нагрузкой и

диагностирования рака молочной железы, DICOM – это стандарт, который позволяет визуализировать и обрабатывать медицинские изображения из самых различных областей здравоохранения [2].

DICOM файл содержит в себе такую мета-информацию, как:

1. Личные данные и параметры пациента.
2. Модель и фирму производителя аппарата, на котором проводилось обследование.
3. Атрибуты медицинского учреждения, где было проведено обследование.
4. Данные о медицинском персонале, проводившем обследование пациента.
5. Вид, время и дата проводимого обследования.
6. Условия и параметры проводимого исследования.
7. Параметры изображения или серии изображений, записанных в DICOM-файле.
8. Уникальные ключи идентификации *Unique Identifier* (*UID*) групп данных, описанных в DICOM-файле.
9. Изображение или серию изображений, полученных при обследовании пациента.
10. Представление PDF-документов в DICOM-файле.
11. Представление DICOM-записи на оптические носители. [4].

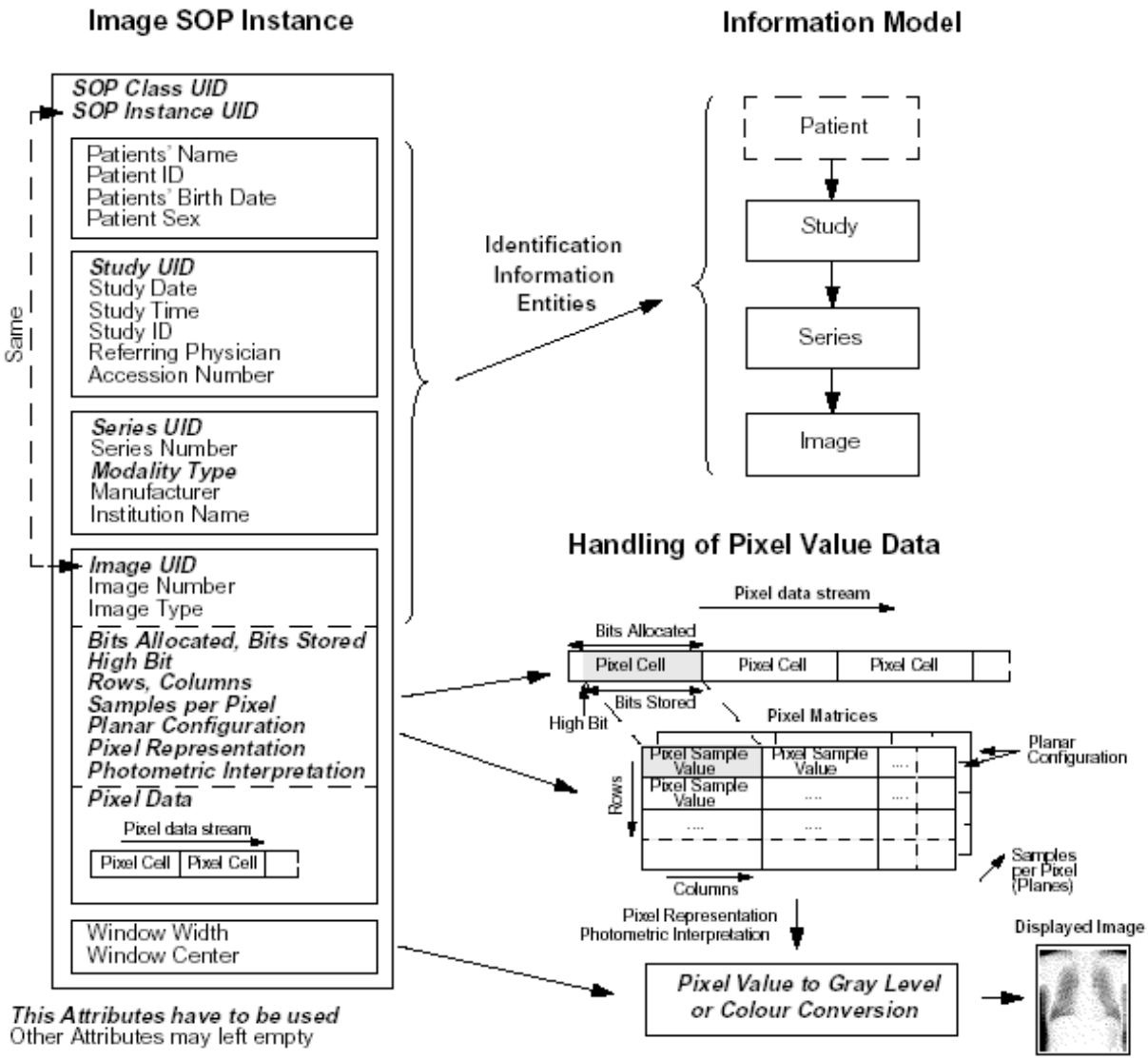

Рис. 1.1 Структура данных формата DICOM

Обзор существующих аналогов

На сегодняшний день существует множество приложений для визуализации DICOM изображений. Каждое из них обладает как определенными достоинствами, так и определенными недостатками.

Приложения бывают различной направленности и могут реализовывать самые различные цели: одни созданы под работу в браузере для которых необходимо наличие интернета, другие — автономные приложения, для работы которых не требуется доступ в интернет. В данном разделе будут рассмотрены только автономные приложения.

Самыми известными и широко используемыми программами для визуализации медицинских данных являются RadiAnt, OsiriX и MRIcro. В этом разделе будут рассмотрены достоинства и недостатки существующих приложений в сравнении с приложением, реализованным в ходе данной работы.

1. RadiAnt

Это простое в использовании, понятное любому пользователю приложение, содержащее минимальный набор инструментов для визуализации DICOM изображений [6].

Достоинства:

- Больше инструментов для различных измерений: угол между двумя прямыми, возможность обвести произвольный контур для вычисления площади.
- Возможность просмотра двух и более изображений на одном экране.

Недостатки:

- Отсутствие палитр, применяемых для лучшей визуализации различий цветового сигнала в черно-белых изображениях.
- Отсутствие русскоязычного интерфейса.

2. OsiriX

Простое в использовании, интуитивно понятное приложение для устройств Mac OS, содержащее большое количество инструментов для визуализации и обработки DICOM изображений [7].

Достоинства:

- Больше инструментов для визуализации.
- Возможность просмотра двух и более изображений на одном экране.

Недостатки:

- Использование возможно только на устройствах с Mac OS.
- Отсутствие палитр, применяемых для лучшей визуализации различий

цветового сигнала в черно-белых изображениях.

– Отсутствие русскоязычного интерфейса.

3. MRIcro

Это приложение для визуализации DICOM изображений, рассчитанное на опытных пользователей. Содержит в себе большое количество инструментов, необходимых для визуализации и обработки изображений [8].

Достоинства:

– Больше различных инструментов для визуализации.

Недостатки:

– Отсутствие палитр, применяемых для лучшей визуализации различий цветового сигнала в черно-белых изображениях.
– Не интуитивно понятный интерфейс, не удобно для использования неопытным пользователем.
– Отсутствие русскоязычного интерфейса.

Для удобства сравнения описанных приложений рассмотрим следующую таблицу:

	RadiAnt	OsiriX	MRIcro	Данное приложение
Расширенный набор инструментов визуализации	-	-	+	-
Расширенный набор инструментов для измерений	+	+	-	-
Наличие палитр	-	+	-	+
Кросс-платформенность	+	-	+	+
Наличие русскоязычного интерфейса	-	-	-	+
Удобный пользовательский интерфейс	+	+	-	+

Таблица 1. Сравнение приложений для визуализации DICOM изображений

Основные требования

После изучения стандарта DICOM и обзора существующих приложений для визуализации медицинских данных были сформулированы основные требования к реализуемому программному продукту:

1. Загрузка и визуализация DICOM изображения.
2. Изменение размера окна для черно-белых изображений.
3. Применение различных базовых таблиц цветов для черно-белых изображений.
4. Изменение размера изображения, инвертация цветов, поворот изображения.
5. Возможность просмотра нескольких исследований и серий изображений.
6. Измерения расстояний на изображении.
7. Измерение площади выделенной части изображения.
8. Вывод значений всех тегов изображения.
9. Возможность поиска и загрузки изображений с PACS-сервера.

Загрузка и визуализация DICOM изображения

Самым первым шагом в визуализации DICOM изображений является загрузка самого DICOM файла и считывание из него всех необходимых для правильной визуализации значений тегов. Основные теги, необходимые для корректной визуализации: *Pixel Data, Rows, Columns, Pixel Spacing, Bits Allocated, Rescale Slope, Window Center, Window Width, Modality* и *Photometric Interpretation*. В массиве данных, возвращаемых по тегу Pixel Data способы кодирования цветов отличны от стандартного промежутка 0 до 255. В DICOM этот промежуток может варьироваться, в зависимости от параметров изображения. Следовательно, для корректной визуализации DICOM изображения необходимо построить линейное преобразование промежутка для DICOM в соответствующий промежуток *(0; 255)*, и применить это преобразование ко всем пикселям DICOM изображения.

Изменение размера «окна»

Для выделения частей изображения с различной плотностью часто используется возможность интерактивно изменять границы диапазона значений пикселей — «окно»: все пиксели со значением меньше нижней границы «окна» — черные, а пиксели со значением больше верхней границы «окна» — белые. На рисунках 1.2 и 1.3 представлено изображение с различными размерами окна.

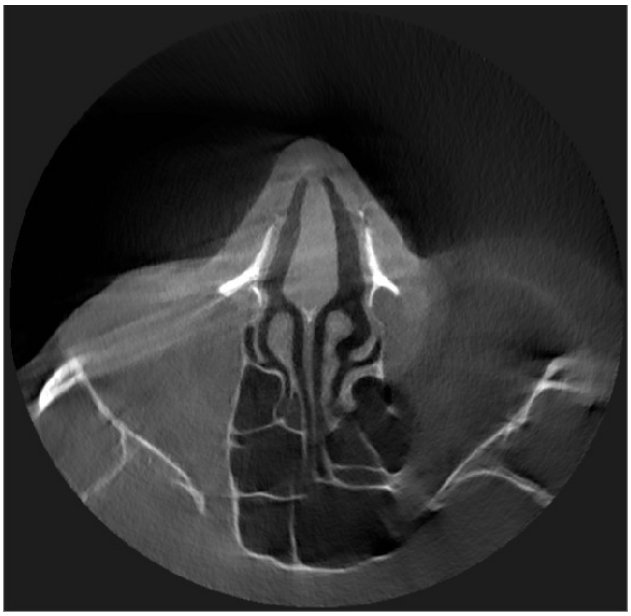

Рис. 1.2 Исходное изображение

Рис. 1.3 Другой размер окна

Применение палитр цветов

Еще одной задачей визуализации DICOM изображений является изменение цветовой палитры черно-белых изображений для того, чтобы различия в интенсивности цветового сигнала были более очевидными для человеческого глаза. В ядерной медицине наиболее известны следующие виды палитр:

1) Палитра «Hot Iron»
2) Палитра «Hot Metal Blue»
3) Палитра «Pet»
4) Палитра «Pet 20 Step»

На рисунках 1.5 — 1.8 изображены результаты применения различных палитр к исходному изображению на рисунке 1.4.

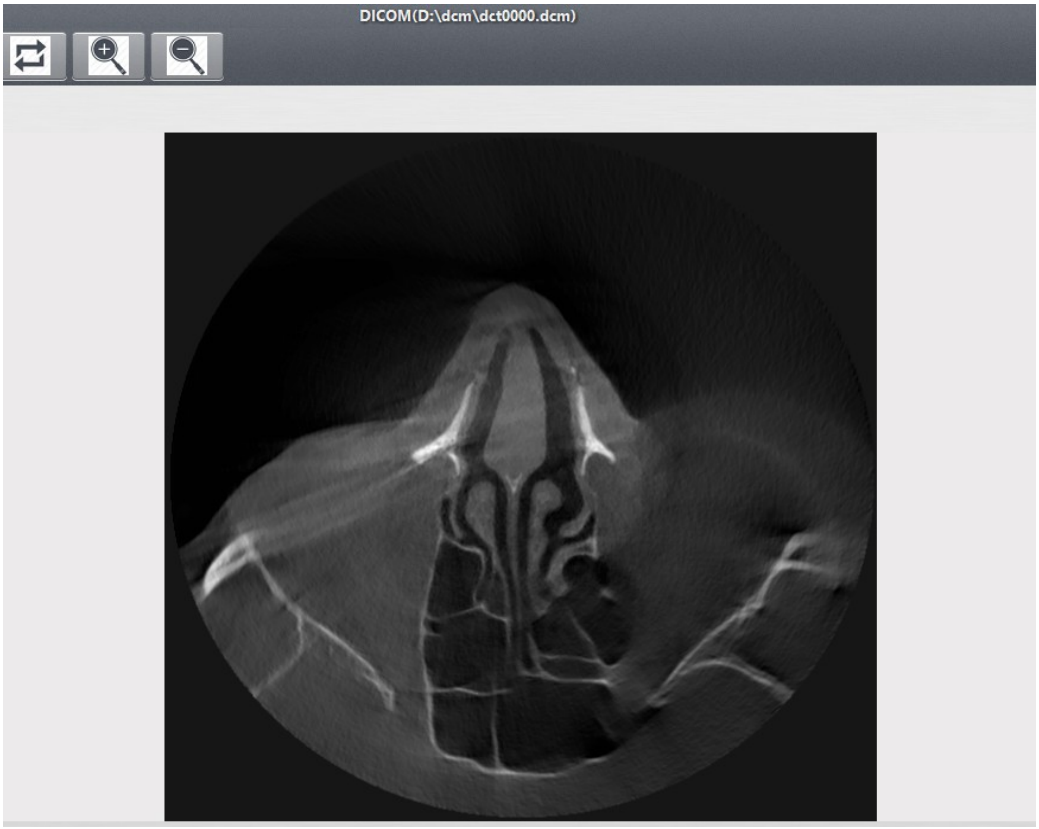

Рис. 1.4 Исходное изображение

Рис. 1.5. Применение к исходному изображению палитры Hot Iron

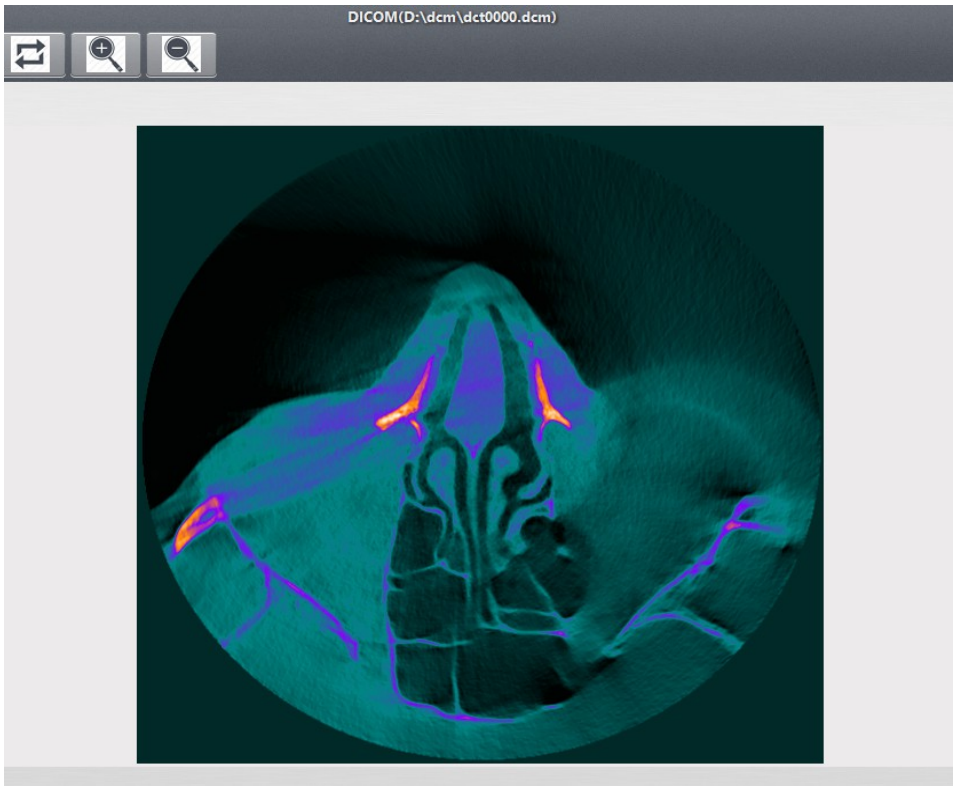

Рис. 1.6. Применение к исходному изображению палитры PET

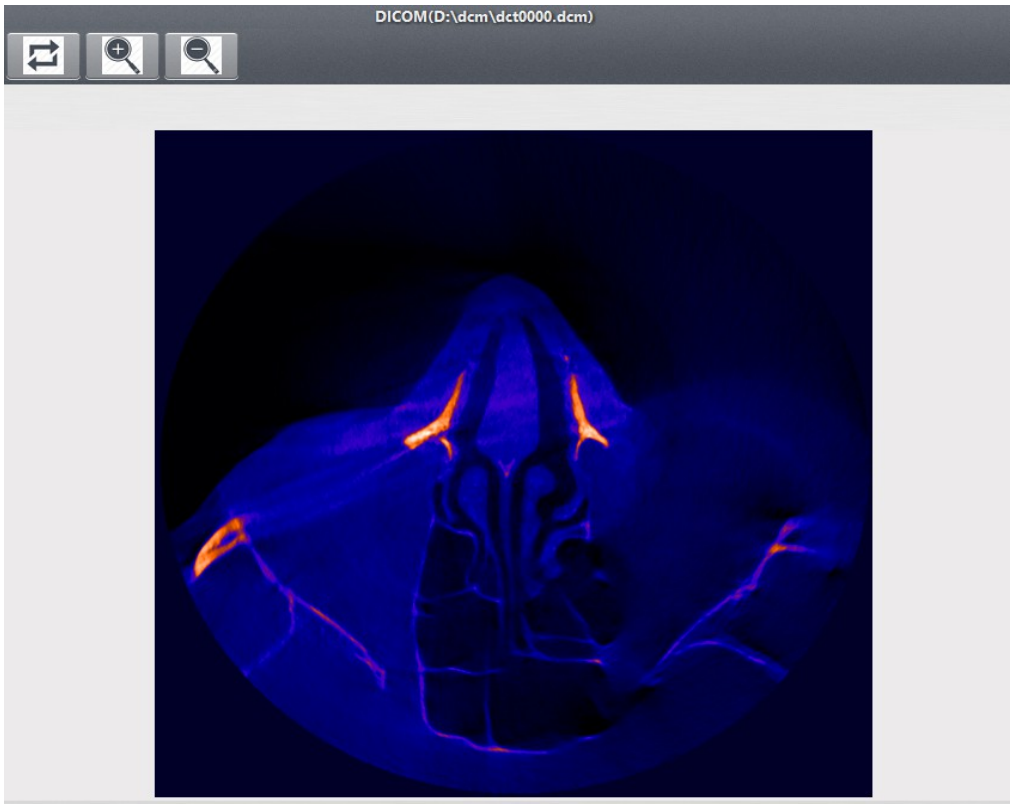

Рис. 1.7. Применение к исходному изображению палитры Hot Metal Blue

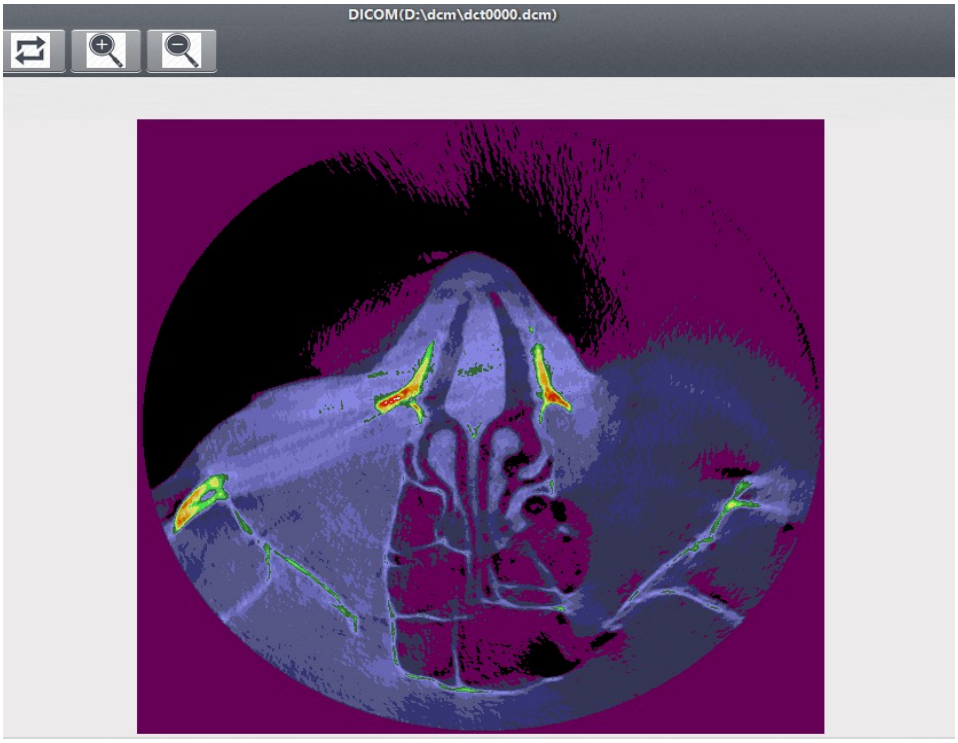

Рис. 1.8. Применение к исходному изображению палитры PET 20 Step

Изменение размера изображения

Еще одним необходимым для врачей, работающих с медицинскими изображениями, требованием является возможность увеличения изображения для более детального рассмотрения отдельных частей.

Инвертация цветов изображения

Позволяет более явно выделить определенные части снимка различной плотности (Рис. 1.9 и Рис. 1.10).

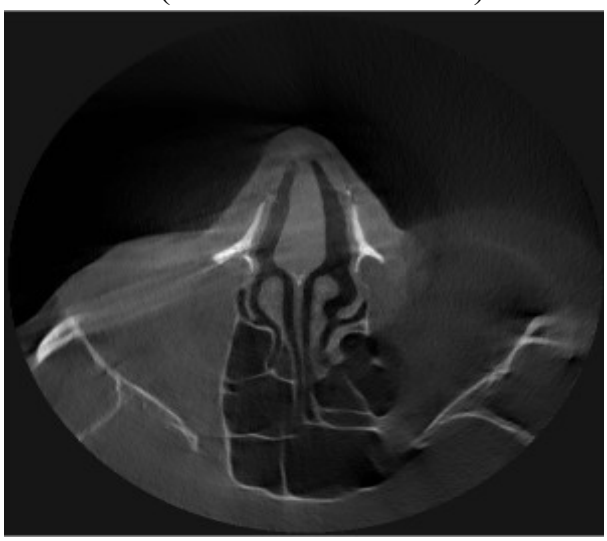

Рис. 1.9. Исходное изображение

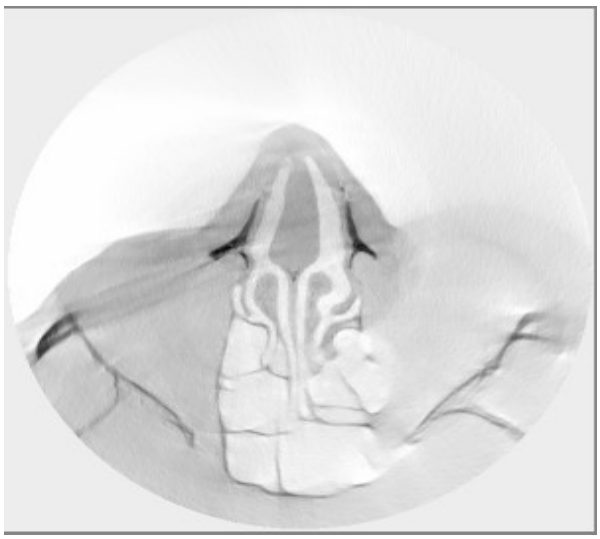

Рис. 1.10. Инвертированное изображение

Просмотр исследований и серий

Одним из основных требований к программе просмотра DICOM изображений является возможность просмотра сразу нескольких исследований и серий изображений. Пользователь может открыть сразу множество файлов, они должны быть сгруппированы по исследованиям и сериям.

Измерение расстояний

Еще одним часто используемым инструментом программ для просмотра DICOM изображений является возможность измерения расстояний на изображении, что позволяет более точно диагностировать различные заболевания. При проведении линии на изображении, должно измеряться расстояние от ее начала до конца и на экран должен выводиться результат (Рис. 1.11).

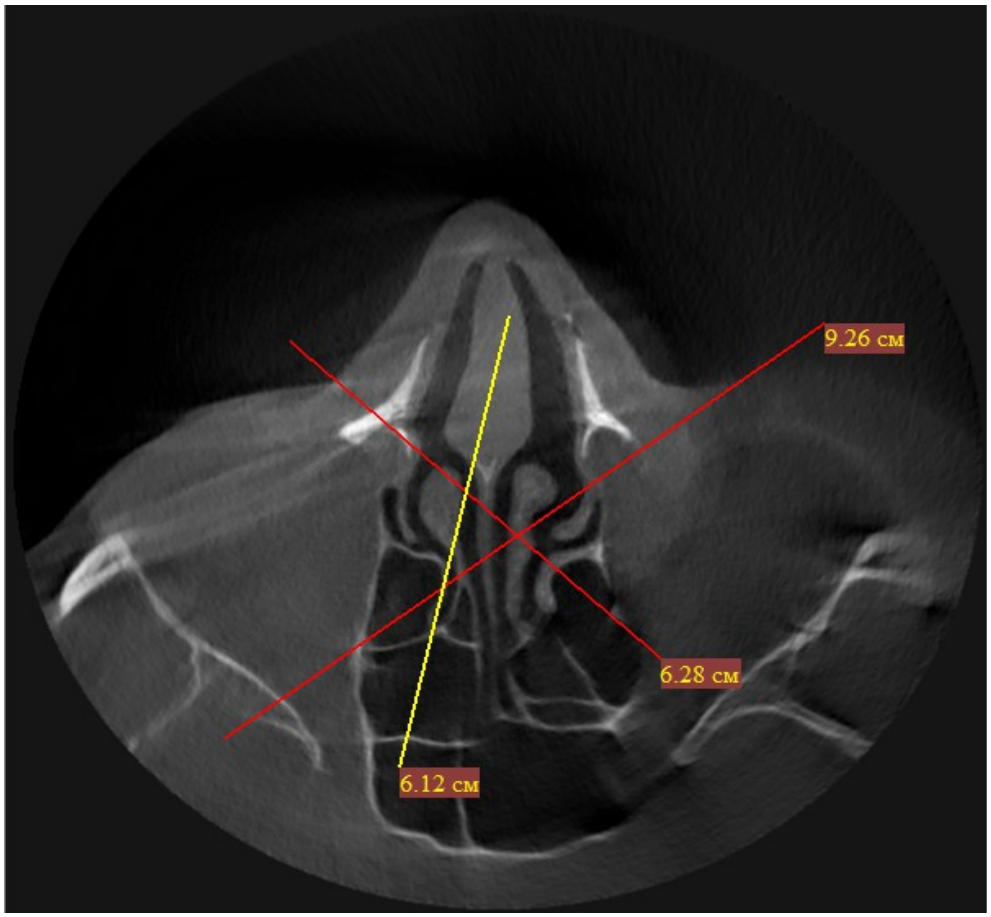

Рис. 1.11. Измерение расстояний

Измерение площади

Помимо измерения длины, врачу может понадобиться измерить площадь или определить максимальное, минимальное или среднее значение интенсивности цветового сигнала в выделенной области. При отрисовке эллипса на изображении на экран должна выводиться информация о площади эллипса, максимальном, минимальном или среднем значении интенсивности цветового сигнала в выделенной области (Рис. 1.12).

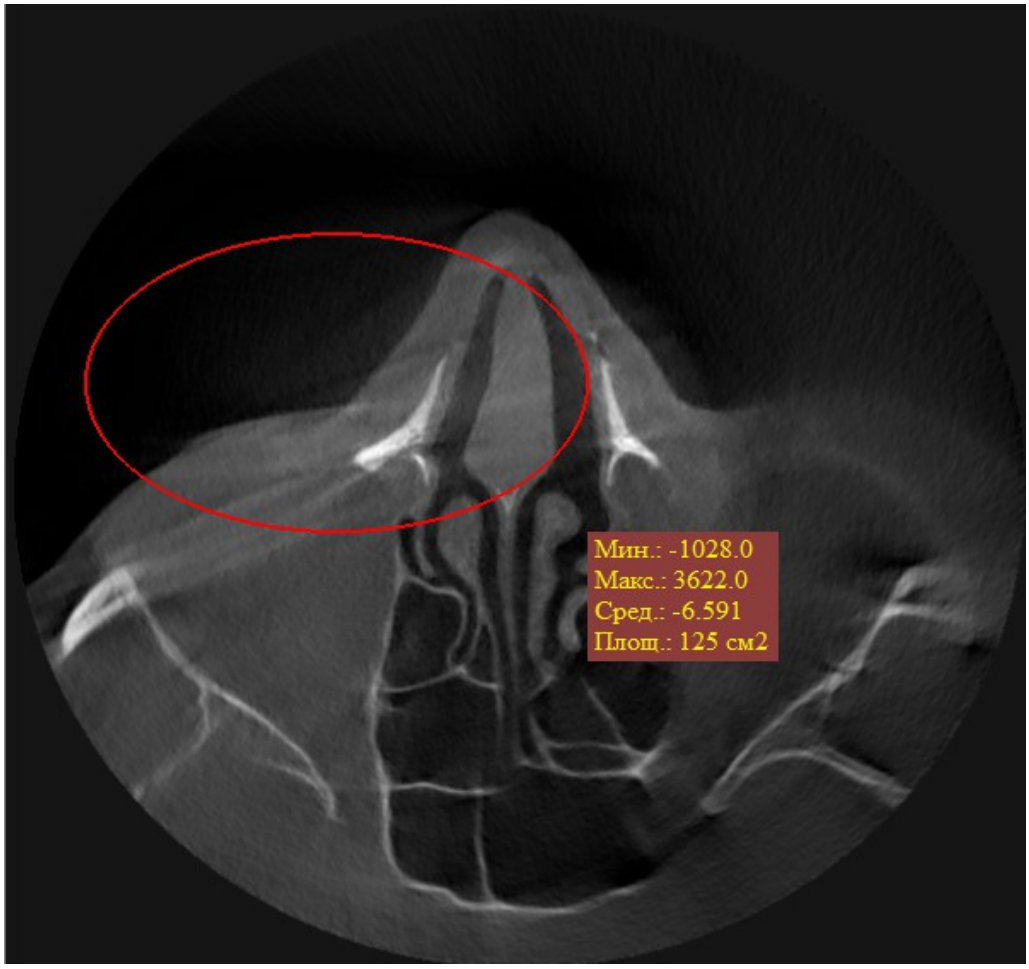

Рис. 1.12. Измерение площади, среднего, минимального и максимального значений эллипса

Сохранение в формате PNG

После визуализации и обработки DICOM изображения может понадобиться сохранить его в другом формате для дальнейшей передачи, обработки либо печати.

Теги DICOM файла

Так как DICOM файл содержит в себе большое количество тегов с различной мета-информацией, у врача может возникнуть необходимость посмотреть значение какого-либо из них, для этого необходимо реализовать возможность просмотра значений всех тегов DICOM файла.

Взаимодействие с PACS-сервером

Стандарт DICOM поддерживает сервис Query/Retrive — сервис для запроса/получения списка пациентов или исследований с других DICOM устройств.

PACS — системы передачи и архивации DICOM изображений, предполагают создание специальных удаленных архивов на DICOM серверах, где весьма объемный архив может длительное время существовать в «горячем» виде и быть быстро доступным для поиска и просмотра интересующей информации по DICOM сети. PACS-системы поддерживают следующие основные операции:

1. C-FIND позволяет производить поиск DICOM изображений на различных уровнях, таких как: *PATIENT, STUDY, SERIES* и *IMAGE.*
2. C-STORE позволяет сохранять изображения формата DCM на PACS-сервере.
3. C-MOVE позволяет передавать изображения из PACS на клиентскую машину. Для передачи данных необходимо настроить соединение с сервером, где будут описываться параметры клиента, на который посылаются данные. На клиенте необходимо запустить PACS, выполняющий команду C-STORE.

Для доступа к PACS-серверу необходимо реализовать возможность установки параметров подключения к серверу: IP адрес, номер порта, название подключения, описание подключения.

2. АРХИТЕКТУРА ПРИЛОЖЕНИЯ

Для реализации данного приложения был выбран язык программирования Java, так как, благодаря тому, что приложения выполняются с помощью виртуальной машины (JVM), язык Java является платформонезависимым.

После анализа требований к программному продукту было решено воспользоваться средствами OpenGL. Данная технология позволяет создавать кроссплатформенные приложения, что означает возможность использовать ПО на различных операционных системах. Одним из важных преимуществ технологии OpenGL является использование шейдеров для обработки данных. Шейдерная программа выполняется полностью на графическом процессоре видеокарты (GPU) [13]. Данный подход значительно ускоряет все расчеты, связанные с изображением, по сравнению с обычной обработкой данных на центральном процессоре (CPU). На сегодняшний день не так много разработчиков приложений для визуализации DICOM изображений используют данный подход, что также является одной из причин выбора этой технологии.

Анализ существующих технологий показал, что для доступа к спецификации OpenGL лучше всего подходит библиотека LWJGL, так как она хорошо документирована, удобна в использовании и содержит все необходимые средства для работы с OpenGL. Для работы непосредственно с DICOM файлами была выбрана библиотека dcm4che, так как она хорошо документирована, удобна в использовании и является одной из основных библиотек на языке java, реализующих стандарт DICOM.

Для реализации графического интерфейса была выбрана технология JavaFX.

Библиотека dcm4che

Библиотека dcm4che это набор приложений и утилит с открытым исходным кодом для разработки приложений в сфере медицинского

программного обеспечения [9]. Разработка библиотеки ведется на языке программирования Java, поскольку это позволяет разрабатывать выскопроизводителные и кросс-платформенные компоненты. Библиотека dcm4che реализует стандарт DICOM.

Это приложение содержит набор интерфейсов и сервисов необходимых для разработки приложений, обеспечивающих хранение, обработку и передачу медицинских изображений.

Технология JavaFX

JavaFX представляет собой набор графических и медиа-пакетов, что позволяет разработчикам проектировать, создавать, тестировать, отлаживать и развертывать насыщенные клиентские приложения (RIA), которые работают для различных платформах [10].

Билиотека LWJGL

Lightweight Java Game Library (LWJGL) — открытая графическая библиотека, основной целью которой является предоставление простого и легковесного программного интерфейса для создания компьютерных игр на языке Java [11].

LWJGL представляет собой библиотеку Java, которая позволяет получить доступ к популярным встроенным программным интерфейсам, используемым в разработке графики (OpenGL), аудио (OpenAL) и приложений с параллельными вычислениями (OpenCL). Эта библиотека дает прямой доступ к спецификации OpenGL, что делает программы, написанные на ней высокопроизводительными.

Технология OpenGL

OpenGL (**Open G**raphics **L**ibrary) — спецификация, определяющая платформонезависимый (независимый от языка программирования)

программный интерфейс для написания приложений, использующих двумерную и трёхмернуюкомпьютерную графику.

С момента своего появления в 1992 году, OpenGL стал наиболее широко используемым программным интерфейсом (API) в отрасли 2D и 3D грфики, позволяя создавать программное обеспечение для большого количества компьютерных платформ. OpenGL способствует инновациям и ускоряет разработку приложений за счет включения широкого спектра различных функциональных возможностей, таких как: рендеринг, наложение текстур, спецэффектов и многих других функций компьютерной визуализации. Разработчики могут использовть OpenGL на различных платформах, что обеспечивает широкое развертывание приложения [14].

GLSL Шейдеры

Шейдер является небольшой программой, написанной на специальном языке высокого уровня для программирования шейдеров GLSL (OpenGL Shading Language), которая заменяет собой часть графического конвейера видеокарты. Тип шейдера зависит от того, какую часть конвейера он замещает. В данной работе используются два типа шейдеров: вершинный и фрагментный [12].

Вершинный шейдер замещает часть графического конвейера, отвечающую за преобразования, связанные с положением вершины. В данном шейдере определяется позиция вершины и результат записывается в переменную gl_Position.

Фрагментный шейдер замещает часть графического конвейера, отвечающую за преобразования, связанные с цветом вершины. В данном шейдере, исходя из различных параметров текстур и освещения определяется цвет вершины и результат записывается в переменную gl_FragColor.

3. РЕАЛИЗАЦИЯ

В данном разделе будут описаны подходы к реализации некоторых функциональных возможностей, описанных в предыдущих разделах.

Изображение в DICOM файле хранится в виде массива данных, которые могут быть различного размера (Byte, Short) и типа (unsigned, signed). Так же, в самом файле хранятся значения ширины и высоты исходного изображения. Таким образом, изображение можно представить в виде матрицы чисел различной размерности. Для вывода на экран полученного изображения, оно передается во фрагментный шейдер, где все значения преобразуются в интервал *(0; 255)*. В вершинном шейдере определяется позиция изображения на экране.

1. Матрицы трансформацй

Для реализации трансформаций изображения, таких как: поворот, увеличение и уменьшение размера, сдвиг используются специальные матрицы Модели (Model), Вида (View), Проекции (Projection). Эти матрицы позволяют реализовать трансформацию объекта в пространстве [15].

Все трансформации применяются к позициям элементов изображения (вершинам). Все вершины представлены в виде векторов (x, y, z, w), где *x, y, z* – координаты вершины в соответствующих осях, а *w* – всегда принимает значение 1.

1. Матрица перемещений.

Пусть *X, Y, Z* – значения, на которые необходимо передвинуть вершину по осям *x, y* и *z* соответственно. Следовательно, матрица перемещения будет иметь следующий вид:

$$M_{transalation} = \begin{bmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Для применения матрицы перемещения необходимо умножить матрицу на

исходный вектор (x, y, z, w) :

$$\begin{bmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x+X \\ y+Y \\ z+Z \\ 1 \end{bmatrix}$$

2. Матрица масштабирования.

Пусть необходимо увеличить вектор в *X, Y* и *Z* раз по соответствующим направлениям. Следовательно, матрица масштабирования будет иметь следующий вид:

$$M_{scale} = \begin{bmatrix} X & 0 & 0 & 0 \\ 0 & Y & 0 & 0 \\ 0 & 0 & Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Для применения матрицы масштабирования необходимо умножить матрицу на исходный вектор (x, y, z, w) :

$$\begin{bmatrix} X & 0 & 0 & 0 \\ 0 & Y & 0 & 0 \\ 0 & 0 & Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x*X \\ y*Y \\ z*Z \\ 1 \end{bmatrix} .$$

3. Матрица вращения.

Вращение задается углом и осью, относительно которой будет происходить поворот.

Матрица вращения вокруг оси *Z* на угол α имеет следующий вид:

$$M_{rotation} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Для применения матрицы вращения необходимо умножить матрицу на исходный вектор (x, y, z, w) :

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x*\cos(\alpha) - y*\sin(\alpha) \\ x*\sin(\alpha) + y*\cos(\alpha) \\ z \\ 1 \end{bmatrix} .$$

Матрица вращения вокруг оси Y на угол α имеет следующий вид:

$$M_{rotation}=\begin{bmatrix}\cos(\alpha) & 0 & -\sin(\alpha) & 0\\ 0 & 1 & 0 & 0\\ \sin(\alpha) & 0 & \cos(\alpha) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

Применение матрицы вращения к исходному вектору:

$$\begin{bmatrix}\cos(\alpha) & 0 & -\sin(\alpha) & 0\\ 0 & 1 & 0 & 0\\ \sin(\alpha) & 0 & \cos(\alpha) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}*\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}=\begin{bmatrix}x*\cos(\alpha)-z*\sin(\alpha)\\ y\\ x*\sin(\alpha)+z*\cos(\alpha)\\ 1\end{bmatrix}.$$

Матрица вращения вокруг оси X на угол α имеет следующий вид:

$$M_{rotation}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & \cos(\alpha) & -\sin(\alpha) & 0\\ 0 & \sin(\alpha) & \cos(\alpha) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

Применение матрицы вращения к исходному вектору:

$$\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & \cos(\alpha) & -\sin(\alpha) & 0\\ 0 & \sin(\alpha) & \cos(\alpha) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}*\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}=\begin{bmatrix}x\\ y*\cos(\alpha)-z*\sin(\alpha)\\ y*\sin(\alpha)+z*\cos(\alpha)\\ 1\end{bmatrix}.$$

Для объединения этих трансформаций необходимо перемножить матрицы и вектор друг на друга в следующем порядке:

$$(x_t, y_t, z_t, 1)=M_{translation}*M_{rotation}*M_{scale}*(x, y, z, 1).$$

В данном случае, матрица преобразований передается в вершинный шейдер, где применяется ко всем вершинам изображения. Вершинный шейдер имеет следующий вид:

```
uniform mat4 transformMatrix;
attribute vec4 in_Position;

void main() {

	gl_Position = transformMatrix * gl_Position;
	gl_TexCoord[0] = gl_MultiTexCoord0;
}
```

2. Преобразования цветов

Исходное изображение может содержать в себе значения, не входящие в интервал *(0; 255)*, который необходим для корректного отображения цветов.

Следовательно, нужно перевести все значения изображения в интервал от 0 до 255. Для этого необходимо определить минимальное (Im_{min}) и максимальное (Im_{max}) значения данных для этого изображения, а затем применить формулу (2.1) ко всем элементам изображения:

$$val_n = \frac{val - \mathrm{Im}_{min}}{\mathrm{Im}_{max} - \mathrm{Im}_{min}} * 255 \quad (2.1)$$

Для реализации изменения окна необходимо все значения, меньшие, чем нижняя граница окна заменить на 0, все значения, большие чем верхняя граница окна заменить на 255, а для остальных, применить формулу (1.1).

Для реализации инвертации цветов достаточно значение каждого элемента изображения, посчитанного по формуле (2.1) преобразовать по формуле (2.2):

$$val_{inv} = 255 - val_n \quad (2.2)$$

3. Применение палитр

Для удобства было принято решение цветовые палитры хранить в виде png изображений размера 1x255. Для применения палитры вычисляется значение каждого элемента изображения по алгоритму из предыдущего пункта, а затем цвет для данного пикселя берется из png изображения из позиции, равной значению этого элемента.

Все преобразования, описанные в пунктах 2-3 данного раздела, реализованы во фрагментом шейдере, который имеет следующий вид:

```
uniform isampler2D texture1;
uniform sampler1D texture2;
uniform int from;
uniform int to;
uniform int width;
uniform int height;
uniform int isUsePalette;
uniform int isInvert;

void main() {
vec4 texColor;
int col = texelFetch(texture1,ivec2(gl_TexCoord[0].s
```

```
*width,gl_TexCoord[0].t*height), 0);

 float resCol;
 if(col < from)
 {
      resCol = 0.0;
 }
 else if(col > to)
 {
      resCol = 1.0;
 }
 else
 {
      resCol = float(col - from)/float(to - from);
 }
 if(isInvert == 1)
 {
      resCol = 1 - resCol;
 }
 if(isUsePalette == 1)
 {
      gl_FragColor = texelFetch(texture2, int(resCol * 255.0), 0);
 }
 else
 {
      gl_FragColor = resCol;
 }
}
```

4. Преобразование координат

Для удобства хранения координат и корректного отображения всех измерений на экране, при изменении размеров, повороте или сдвиге изображения, необходимо производить преобразование из координат изображения в экранные координаты и наоборот. В OpenGL координаты текстуры с изображением находятся в диапазоне *(-1; 1)*.

Для преобразования координат *(x, y)* экрана в координаты изображения необходимо выполнить следующий алгоритм:

1. Получить новые координаты (x_n, y_n) в промежутке *(-1; 1)* из координат экрана.
2. Составить вектор $(x_n, y_n, 0, 1)$.
3. Инвертировать текущую матрицу трансформации для изображения.
4. Перемножить матрицу из пункта 3 с вектором из пункта 2.

Следующий псевдокод на языке java реализует описанный выше

алгоритм:

```
public Point convertToImageCoord(Point screenCoord){
      float x = 2 * (screenCoord.getX() - widthShift) / width - 1;
      float y = 2 * (screenCoord.getY() - heightShift) / height - 1;
      Matrix4f mat = getTransformMatrix();
      Matrix4f matInv = invert(mat);
      Vector4f res = transform(matInv, new Vector4f(x, y, 0, 1));
      return new Point(res.x, res.y);
}
```

Для преобразования координат *(x, y)* изображения в координаты экрана необходимо выполнить следующий алгоритм:

1. Составить вектор *(x, y, 0, 1)*.
2. Перемножить текущую матрицу трансформации для изображения с вектором из пункта 1.
3. Координаты (x_n, y_n) вектора из пункта 2 перевести из промежутка *(-1; 1)* в координаты экрана.

Следующий псевдокод на языке java реализует описанный выше алгоритм:

```
public Point convertToScreenCoord(Point imageCoord){
      float x = imageCoord.getX();
      float y = imageCoord.getY();
      Vector4f res = transform(getTransformMatrix(), new Vector4f(x, y, 0, 1));
      x = (res.x + 1) * width/2 + widthShift;
      y = (res.y + 1) * height/2 + heightShift;
      return new Point(x, y);
}
```

5. Измерение расстояний

Все отрезки, нарисованные с помощью инструмента «Длина», хранятся в виде двух пар значений: координаты начала (x_1, y_1) и координаты конца (x_2, y_2) отрезка. Для измерения длины отрезка *d* используется формула (5.1)

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (5.1)$$

Для реализации возможности выбора и удаления отрисованного отрезка, необходимо проверить при обработке нажатия клавиши мыши, выбран ли какой-либо отрезок. Для этого необходимо посчитать расстояние от точки на экране до ближайшего отрезка. Если это расстояние меньше, чем какая-то наперед заданная константа ε_{max}, то можно считать, что при данном нажатии

этот отрезок был выбран и он должен быть окрашен в другой цвет. Расстояние от точки до отрезка вычисляется по формуле (5.2).

$$\frac{|(y_2-y_1)x_0-(x_2-x_1)y_0+x_2*y_1-y_2*x_1|}{\sqrt{(y_2-y_1)^2+(x_2-x_1)^2}} \quad (5.2)$$

где (x_0, y_0) — координаты точки, (x_1, y_1) — координаты начала, (x_2, y_2) — координаты конца отрезка.

6. Инструмент «Эллипс»

Эллипсы, нарисованные с помощью данного инструмента хранятся в виде координат центра (x_0, y_0) и двух радиусов a и b. Для подсчета среднего, минимального и максимального значений внутри описанного эллипса необходимо пройти по всем исходным значениям изображения, преобразовать их (по алгоритму из пункта 4 данного раздела) в экранные координаты и проверить, находится ли это значение внутри эллипса. Так как эллипс описывается каноническим уравнением (6.1),

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1 \quad (6.1)$$

следовательно, принадлежность точки (x, y) к эллипсу можно вычислять по формуле (6.2)

$$\frac{(x-x_0)^2}{a^2}+\frac{(y-y_0)^2}{b^2}<1 \quad (6.2)$$

Все значения, принадлежащие описанному эллипсу, необходимо учитывать при подсчете среднего, максимального и минимального значений.

Площадь эллипса определяется по формуле (6.3)

$$S^2=\pi\, ab \quad (6.3)$$

Следующий псевдокод на языке java реализует подсчет описанных выше параметров:

```
/**
      * @param img - исходное изображение
      * @param elCenter - центр эллипса
      * @param a, b - радиусы
*/
```

```
public Map<EllipseParam, Float> calculateEllipseParams(DicomImage img, Point
elCenter, float a, float b){
      Object[] buffer = img.getImageBuffer();
      for(float y = 0; y < img.getHeight(); y++)
            for(float x = 0; x < img.getWidth(); x++){
                  Object elem = buffer[(int)y * img.getWidth() + (int)x];
                  float val = getValue(elem);
                  Point screenCoord = convertToScreenCoord(x, y);
                  if(isPointInEllipse(elCenter, a, b, screenCoord){
                        resMin = val < resMin ? val : resMin;
                        resMax = val > resMax ? val : resMax;
                        resMean += val;
                        counter++;
                  }
            }
      resArea = (float)round(3.1415926f * Math.abs(a) * Math.abs(b)/100);
      resMean /= counter;
      result.put(EllipseParam.AREA, resArea);
      result.put(EllipseParam.MAX, resMax);
      result.put(EllipseParam.MIN, resMin);
      result.put(EllipseParam.MEAN, resMean);
      return result;
}
```

Для реализации возможности выбора и удаления отрисованного эллипса, необходимо проверить при обработке нажатия клавиши мыши, выбран ли какой-либо эллипс. Для этого задается константа ε_{max}, обозначающая максимальную погрешность в выборе эллипса. Чтобы точка на экране считалась принадлежащей границе эллипса, необходимо, чтобы одновременно выполнялись условия (6.4).

$$\begin{cases} \dfrac{(x-x_0)^2}{(a+\varepsilon_{max})^2}+\dfrac{(y-y_0)^2}{(b+\varepsilon_{max})^2}<1 \\ \dfrac{(x-x_0)^2}{(a-\varepsilon_{max})^2}+\dfrac{(y-y_0)^2}{(b-\varepsilon_{max})^2}>1 \end{cases} \tag{6.4}$$

7. Группировка по исследованиям и сериям

С левой стороны экрана находится панель, в которой расположены исследования, внутри которых находятся серии изображений. Для исследований и серий выводятся их названия. (Рис. 3.1).

При нажатии на серию изображений откроется первое изображение серии, просмотр дальнейших изображений серии возможен либо с помощью прокрутки колесика мыши, при выборе значка «Серия кадров» в панели инструментов, либо с помощью ползунка, находящегося в левой стороне экрана.

При этом на экране отображается общее количество изображений в серии и номер текущего.

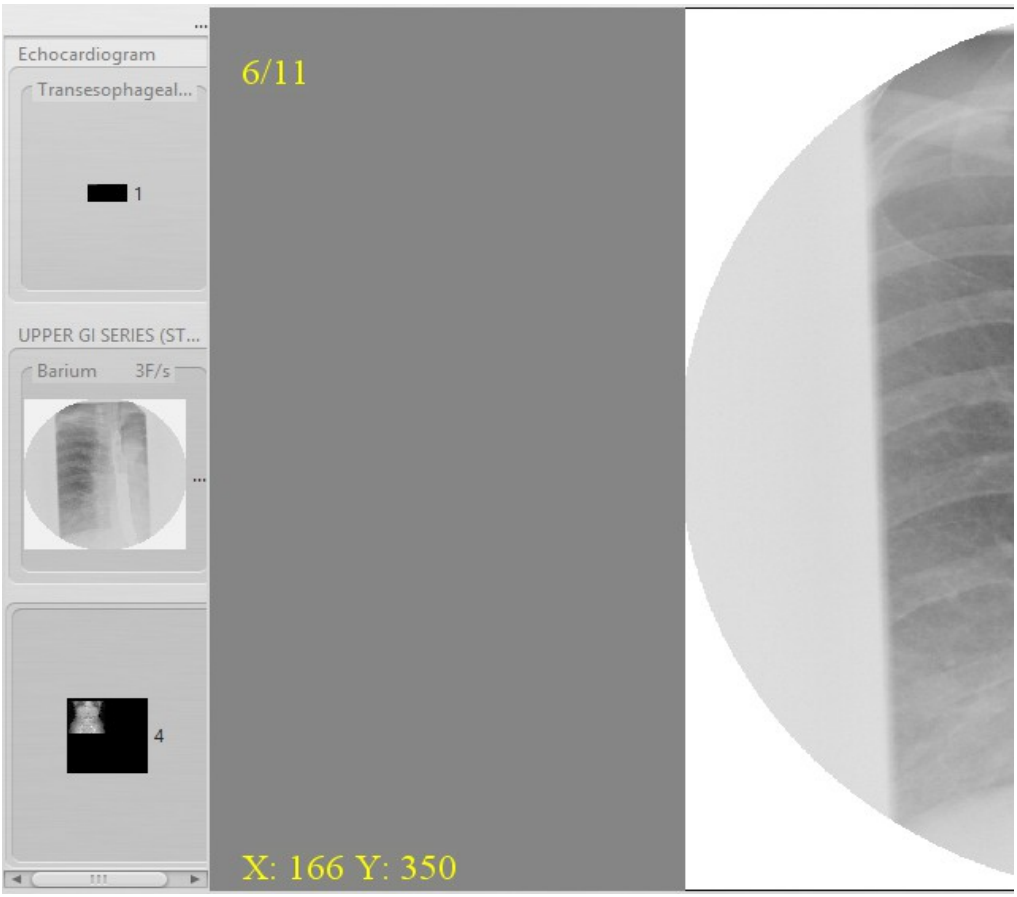

Рис. 3.1 Панель исследований и серий изображений

8. Теги DICOM файла

При выборе в панели инструментов «Файл» → «Информация» открывается диалог с возможностью прокрутки со значениями всех тегов (Рис. 3.2). Значения всех тегов доступны с помощью библиотеки dcm4chee.

9. Взаимодействие с PACS-сервером

Для доступа к PACS-серверу необходимо выбрать пункт меню «Загрузить с PACS-сервера» в панели инструментов. После этого на экран будет выведен диалог с возможностью установки параметров подключения к серверу: IP адрес, номер порта, название подключения, описание подключения (Рис. 3.3).

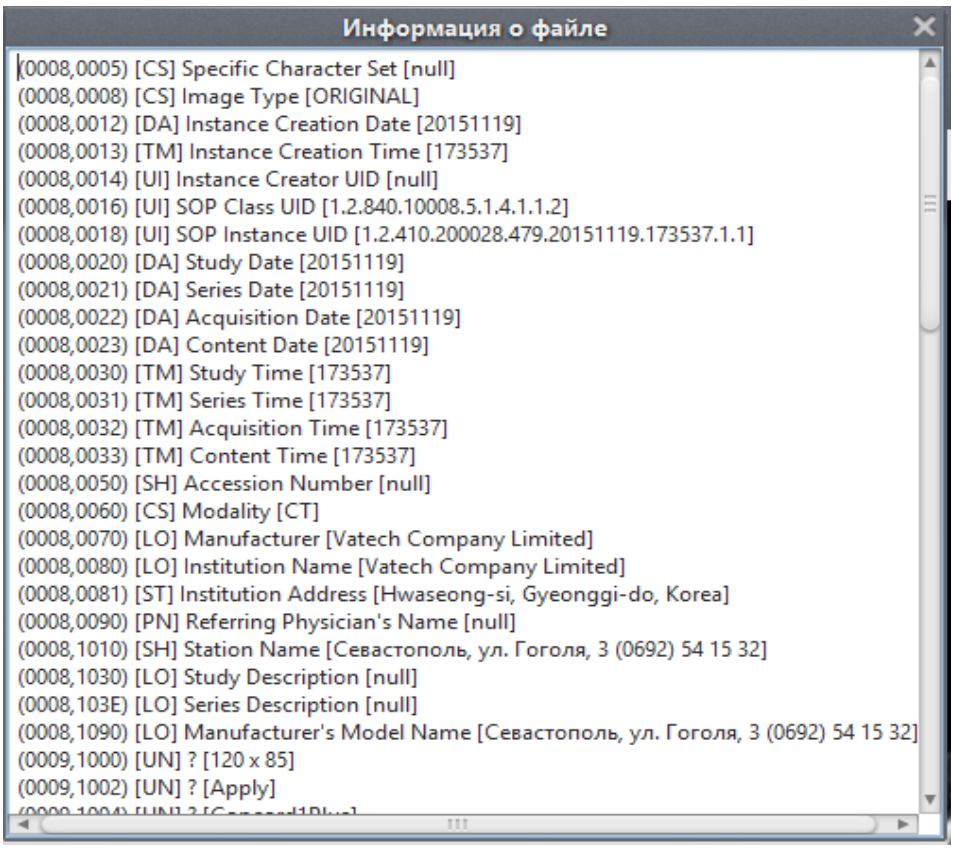

Рис. 3.2. Диалог со значениями тегов

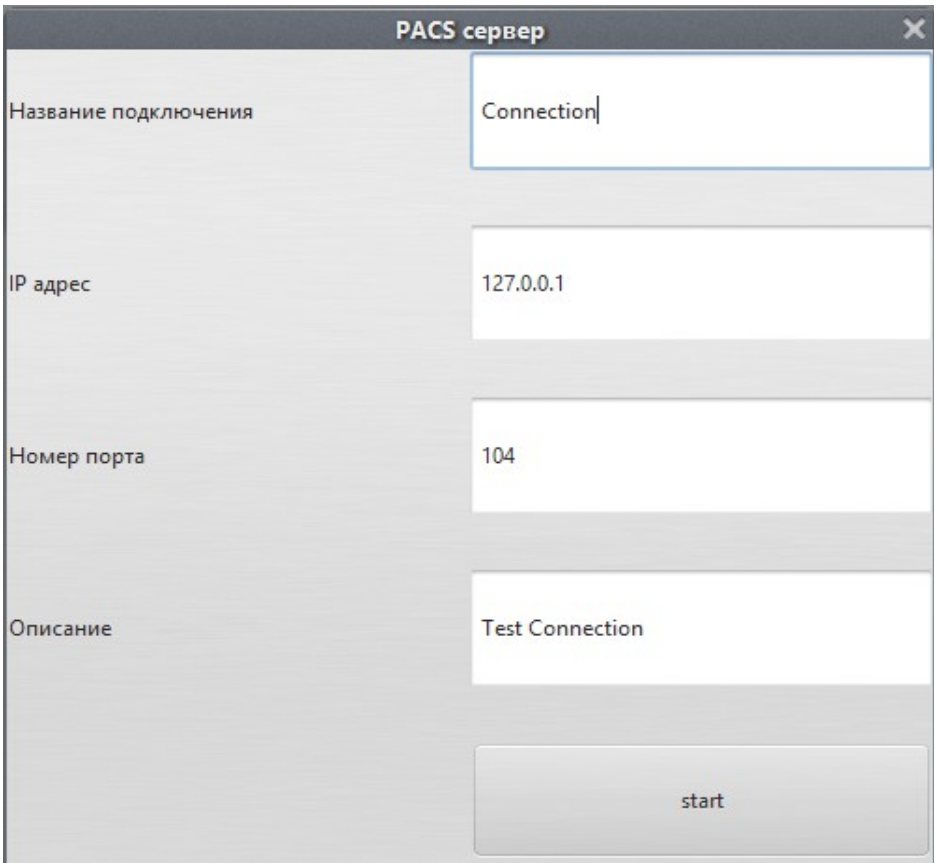

Рис. 3.3. Диалог подключения к PACS-серверу

10. Развертывание приложения

Исходные коды и бинарный файл приложения находятся по адресу https://github.com/teplovas/dicom_project/.

Для развертывания приложения необходимо:

1. Скачать файл DicomViewer.jar по указанному адресу .
2. Запустить его.

Для запуска данного приложения на рабочей машине необходимо наличие JRE версии 1.8. JRE – это среда для выполнения Java-приложения, состоящая из JVM и необходимых Java-библиотек.

4. ТЕСТИРОВАНИЕ

Тестирование всех реализованных функциональных возможностей проводилось на большом количестве DICOM изображений из разных исследований (study) и серий (series), с различной размерностью, модальностью и типом данных.

Виды *модальности*, на которых проводилось тестирование [16]:

- RF (Radio Fluoroscopy)
- US (Ultrasound)
- CT (Computed Tomography)
- CR (Computed Radiography)
- DX (Digital Radiography)
- MR (Magnetic Resonance)
- XA (X-Ray Angiography)
- MG (Mammography)
- PT (Positron emission tomography (PET))
- NM (Nuclear Medicine)
- RG (Radiographic imaging (conventional film/screen))
- PX (Panoramic X-Ray)
- ES (Endoscopy)
- XC (External-camera Photography)
- OT (Other)

Тип данных (размер):

– Short

– Byte

Тип данных

– Знаковые (signed)

– Беззнаковые (unsigned).

Тест-кейсы

Были выполнены следующие тест-кейсы для проверки корректности и адекватности работы программы.

Тест-кейс № 1. Загрузка изображения.

Шаги:

- В панели инструментов выбрать «Файл».
- В открывшемся выпадающем списке выбрать пункт «Открыть».
- В появившемся окне выбрать DICOM файл.

Ожидаемый результат:

- DICOM файл корректно отображается, нет видимых артефактов.

Тест-кейс № 2. Информация о тегах DICOM файла.

Шаги:

- Выбрать в панели инструментов «Файл».
- В открывшемся выпадающем списке выбрать пункт «Информация о файле».

Ожидаемый результат:

- Все значения тегов корректно отображаются.

Тест-кейс № 3. Применение палитр.

Шаги:

- Выбрать в панели инструментов «Палитра».
- В открывшемся выпадающем списке выбрать тип палитры.

Ожидаемый результат:

- Цвета изображения соответствуют выбранной палитре.
- При смене типа палитры цвета изображения соответственно меняются.
- При выборе типа палитры «Черно-белая» все цвета возвращаются к исходным.
- Нет видимых артефактов на изображении.

Тест-кейс № 4. Инвертация цветов.

Шаги:

- Выбрать в панели инструментов «Инвертировать цвета».

Ожидаемый результат:

- Все цвета изображения изменились на противоположные.
- При повторном нажатии «Инвертировать цвета» все цвета возвращаются к исходным.

Тест-кейс № 5. Поворот на 90°.

Шаги:

- Выбрать в панели инструментов «Поворот на 90°».

Ожидаемый результат:

- При каждом нажатии изображение поворачивается на 90° вправо.

Тест-кейс № 6. Изменение размеров изображения.

Шаги:

- Выбрать в панели инструментов «Увеличить»/«Уменьшить».

Ожидаемый результат:

- Размер изображения меняется в соответствии с выбранным действием.
- При выборе двух противоположных действий подряд размер изображения остается неизменным.

Тест-кейс № 7. Изменение размеров окна.

Шаги:

- На правой панели сдвинуть «ползунки» верхнего и нижнего значений окна.

Ожидаемый результат:

- Установленные размеры окна применены к исходному изображению.

Тест-кейс № 8. Измерение длины.

Шаги:

- В панели инструментов выбрать «Измерения».
- В открывшемся выпадающем списке выбрать пункт «Длина».
- Провести линию.

Ожидаемый результат:

- Рядом с отрисованной линией появляется ее длина в сантиметрах.
- При нажатии на линию она окрашивается в желтый цвет.
- При нажатии на клавишу Delete на клавиатуре все линии, окрашенные в желтый цвет, удаляются.

Тест-кейс № 9. Измерение эллипса.

Шаги:

- В панели инструментов выбрать «Измерения».
- В открывшемся выпадающем списке выбрать пункт «Эллипс».
- Нарисовать эллипс.

Ожидаемый результат:

- Рядом с эллипсом появляется информация о минимальном, максимальном, среднем значениях в указанном эллипсе, а также его площадь в сантиметрах квадратных.
- При нажатии на эллипс он окрашивается в желтый цвет.
- При нажатии на клавишу Delete на клавиатуре все эллипсы, окрашенные в желтый цвет, удаляются.

Тест-кейс № 10. Объединение измерений.

Шаги:

- Выполнить шаги, описанные в тест-кейсах 6 — 7.

Ожидаемый результат:

- Все пункты выполняются одновременно для измерений «Длина» и «Эллипс».

Тест-кейс № 11. Сохранение в формате png.

Шаги:

- В панели инструментов выбрать «Сохранить в файл».
- В открывшемся окне выбрать папку в которую необходимо сохранить сконвертированный файл.
- Ввести название для нового файла, нажать кнопку Save.

Ожидаемый результат:

- В указанной папке появился новый файл формата png.
- При применении различных преобразований к изображению, в png файл сохраняется изображение со всеми изменениями.

Тест-кейс № 12. Загрузка с PACS-сервера.

Шаги:

- В панели инструментов выбрать «Загрузить с PACS-сервера».
- Ввести параметры подключения, нажать кнопку Start.
- Загрузить файл с удаленного сервера.

Ожидаемый результат:

- Выбранный файл корректно отображается и работа с ним не отличается от файла, загруженного по шагам из тест-кейса № 1.

Тест-кейс № 13. Группировка по исследованиям и сериям.

Шаги:

- Выбрать пункт меню «Файл».
- В открывшемся выпадающем списке выбрать пункт «Открыть».
- В появившемся окне выбрать несколько DICOM файлов из разных исследований и серий.

Ожидаемый результат:

- На левой панели появились миниатюры, соответствующие различным сериям изображений.
- Серии сгруппированы по исследованиям.
- При нажатии на миниатюру серии открывается первое изображение серии.

- При выборе в панели инструментов «Серия кадров», при прокрутке колесика мыши происходит смена изображения на следующее в серии.
- Смена изображения также происходит при сдвиге «ползунка» на правой панели.

Все указанные тест-кейсы были проверены на различных операционных системах.

Таким образом, в ходе тестирования были проверены все описанные выше тест-кейсы. Все тесты для данного приложения были пройдены успешно.

ЗАКЛЮЧЕНИЕ

Таким образом, в ходе данной работы было создано кросс-платформенное, легковесное, простое в использовании, не требующее дополнительного развертывания окружения для установки, приложение для визуализации DICOM изображений. Были реализованы следующие функциональные возможности, отвечающие основным требованиям пользователей:

1. Загрузка и визуализация DICOM изображения.
2. Изменение размера окна для черно-белых изображений.
3. Применение различных базовых таблиц цветов для черно-белых изображений.
4. Изменение размера изображения, инвертация цветов, поворот изображения.
5. Возможность просмотра нескольких исследований и серий изображений.
6. Измерения расстояний на изображении.
7. Измерение площади выделенной части изображения.
8. Вывод значений всех тегов изображения.
9. Возможность поиска и загрузки изображений с PACS-сервера.

Все функциональные возможности приложения были протестированы в соответствии с указанными требованиями.

ВЫВОДЫ

Разработка приложений для визуализации медицинских данных — очень важная отрасль в современной компьютерной диагностике. С развитием медицинской диагностики перед разработчиками программного обеспечения возникают все новые и новые задачи. В настоящее время большое значение имеет создание российского ПО для работы с медицинскими данными.

В данной работе было создано русскоязычное приложение с открытым кодом, отвечающее всем основным требованиям.

В дальнейшем планируется расширять функциональные возможности данного приложения. Следующим необходимым инструментом визуализации является возможность работы с 3D изображениями, создание 3D изображений из серии 2D изображений.

СПИСОК ЛИТЕРАТУРЫ

1. Арлычев М. А., Новиков В. Л., Сидоров А. В., Фиалковский А. М., Котина Е. Д., Овсянников Д. А., Плоских В. А. Двухдетекторный однофотонный эмиссионный томограф «ЭФАТОМ» // Журнал технической физики, 2009. Т. 79, Вып. 10. С. 138 — 146.

2. Стандарт DICOM. http://dicom.nema.org/

3. Shiroma, J. T. (2006). An introduction to DICOM // Veterinary Medicine, P. 19– 20.

4. Mustra Mario, Delac Kresimir, Grgic Mislav. Overview of the DICOM Standardt // ELMAR, 2008. 50th International Symposium. Zadar, Croatia. P. 39–44.

5. Kimura M, Ohe K, Yoshihara H, Ando Y, Kawamata F, Tsuchiya F, Furukawa H, Horiguchi S. MERIT-9: A patient information exchange guideline using MML, HL7 and DICOM // International Journal of Medical Informatics, 1998. No 51. P. 59–68.

6. RadiAnt DICOM Viewer. http://www.radiantviewer.com/

7. OsiriX DICOM Viewer. http://www.osirix-viewer.com/

8. MRIcro DICOM Viewer. http://www.cabiatl.com/mricro/mricro/

9. dcm4chee.org. Open Source Clinical Image and Object Management. http://www.dcm4che.org/

10. Oracle. Java Documentation. JavaFX. http://docs.oracle.com/javase/8/javase-clienttechnologies.htm/

11. Lightweight Java Game Library. https://www.lwjgl.org/

12. Рэнди Дж. Рост. OpenGL. Трёхмерная графика и язык программирования шейдеров. Для профессионалов. СПб.:Питер, 2005. 432 с.

13. Ву М., Дэвис Т., Нейдер Дж., Шрайндер Д. OpenGL. Руководство по программированию. Библиотека программиста. СПб.: Питер, 2006. 624 с.

14. Д. Херн, М. Паулин Бейкер. Компьютерная графика и стандарт OpenGL. Изд. 3-е .М.: Вильямс, 2005. 1168 с.

15. Умнов А. Е. Аналитическая геометрия и линейная алгебра. Изд. 3-е. М.: МФТИ, 2011. 544 с.
16. DICOM Modality. https://www.dicomlibrary.com/dicom/modality/