Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой СИИ
_____ Г. М. Цибульский
«____» _____ 2019 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Разработка модуля динамической визуализации 3D данных

09.04.02 Информационные системы и технологии

09.04.02.01 Информационно-управляющие системы

Руководитель	проф., д-р.физ.-мат. наук	Б. С. Добронец
Студент	КИ17-02-1/1М 031726495	Д. Б. Пархоменко
Рецензент	проф., д-р техн. наук	М. Н. Фаворская
Нормоконтролер	проф., д-р.физ.-мат. наук	Б. С. Добронец

Красноярск 2019

Продолжение титульного листа магистерской диссертации по теме «Разработка модуля динамической визуализации 3D данных».

Нормоконтролер Б. С. Добронец

Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ

Заведующий кафедрой СИИ

_____ Г. М. Цибульский

«____» _____ 2019 г.

ГРАФИК
НАПИСАНИЯ И ОФОРМЛЕНИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ
в форме магистерской диссертации

Студент: Пархоменко Данил Борисович.

Группа: КИ17–02–1/1М. Направление: 09.04.02.01 Информационно-управляющие системы.

Тема выпускной квалификационной работы: разработка модуля динамической визуализации 3D данных.

График выполнения выпускной квалификационной работы (ВКР) приведён в таблице 1.

Таблица 1 – График выполнения этапов ВКР

Наименование / содержание этапа	Срок выполнения	Примечания
Анализ предметной области, подбор литературы	До 14 февраля 2018	
Составление плана работы над ВКР	До 14 марта 2018	
Разработка и предоставление на проверку первой главы	До 30 мая 2018	
Разработка и предоставление на проверку второй главы	До 30 сентября 2018	
Работа над экспериментальной частью исследования	До 29 декабря 2018	
Разработка и предоставление на проверку третьей главы	До 31 января 2019	
Доработка ВКР в соответствии с полученными замечаниями	До 8 мая 2019	
Разработка тезисов доклада и подготовка презентации для защиты	До 1 июня 2019	
Согласование с руководителем тезисов доклада и презентации	До 15 июня 2019	
Прохождение нормоконтроля	До 14 июня 2019	
Ознакомление с отзывом и рецензией	До 16 июня 2019	
Завершение ВКР к защите с учётом отзыва и рецензии	До 19 июня 2019	

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка модуля динамической визуализации 3D данных» содержит 66 страниц текстового документа, 32 иллюстрации, 54 использованных источника, 1 приложение.

Объект исследования — эмпирические данные.

ВИЗУАЛЬНАЯ АНАЛИТИКА, ВИЗУАЛИЗАЦИЯ, НАУЧНАЯ ВИЗУАЛИЗАЦИЯ, ВИЗУАЛЬНОЕ ПРЕДСТАВЛЕНИЕ, ДАННЫЕ, МОДЕЛТРОВАНИЕ, АНАЛИЗ МНОГОМЕРНЫХ ДАННЫХ.

Цель исследования — повышение эффективности и качества отображения 3D данных, при помощи динамической визуализации, стереоформата и очков виртуальной реальности..

С этой целью в результате выполнения работы был проведён анализ данной области, выявлена актуальность темы исследования, проведены многочисленные исследования методов, позволяющих визуализировать 3D данные.

На основании результатов, полученных в ходе исследования, был предложен и реализован метод визуализации данных — метод движения контуров среза. Преимущества данного метода в сравнении с уже существующими методами — простота реализации, наглядность, точность результатов. Данный метод применим в областях, где приходится сталкиваться с визуализацией объемных данных (медицина, архитектура и др.).

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Одним из современных эффективных методов анализа различных научных данных является метод компьютерной визуализации этих данных, или иначе – метод научной визуализации, который находит широкое применение как в теоретических, так и экспериментальных исследованиях.

Метод научной визуализации может использоваться для анализа исходных данных различной природы, при этом могут отличаться и цели анализа, а также могут различаться и используемые графические представления этих данных.

Основная задача метода научной визуализации – сделать невидимое видимым. В качестве невидимого могут выступать как абстрактные, так и реальные объекты и явления, непосредственно недоступные человеческому глазу. Например, массивы многомерных данных со сложной структурой.

Массивы многомерных данных и данных больших объемов заключают в себе множество скрытых связей и закономерностей, зачастую недоступных при использовании имеющихся способов визуального анализа. За несколько прошедших десятилетий значительно возрос интерес к проблемам анализа многомерных данных, которые представляют собой данные содержащие информацию о трех или более признаках для каждого объекта. Сложность исследования подобного рода данных заключается в том, что значительная часть закономерностей и связей между различными параметрами изучаемого набора данных скрыта [4]. Таким образом, крайне актуальной задачей является исследование, модификация и симбиоз существующих методов и способов представления и моделирования многомерных данных, позволяющих осуществлять плодотворный визуальный анализ имеющихся наборов данных.

Целью данной магистерской диссертации являлась разработка модуля динамической визуализации данных.

Также, последовательно определялись и ставились задачи, необходимые для достижения цели:

- провести проблемный анализ представленной тематики и исследование на тему необходимости изучения способов визуализации 3D данных;

- изучить теоретические аспекты визуализации данных и предложить собственные варианты решения сложностей визуализации;

- провести тестирование предложенных методов и выявить наиболее перспективный из них для дальнейшей разработки;

- разработать программный модуль визуализации 3D данных.

В итоге магистерская диссертация представляет собой 3 главы:

- исследование предметной области:

- технологии стерео визуализации изображений;

- разработка программного модуля 3D визуализации данных.

Глава 1 Исследование предметной области

1.1 Проблема и актуальность темы исследования

Человеческая природа такова, что наше постижение мира, накопление знаний о нем, решение задач, возникающих перед человеком возможно двумя различными путями. Один из них позволяет работать с абстрактными цепочками символов, с текстами и т. п. Этот механизм мышления человеческого сознания обычно называют символическим, алгебраическим или логическим. Второй механизм мышления обеспечивает работу с чувственными образами и представлениями об этих образах. Его называют образным, геометрическим, интуитивным и т. д. [3].

Стремление человека выразить мысль, передать идею в форме графического изображения можно отметить еще с самых ранних времен. Графические изображения (на камне, холсте, бумаге, металле и других средствах отображения информации) давно используются в обучении. Экран компьютера как средство пассивного отображения графики не обладает принципиальной новизной. Применение графики в исследовательских работах не только увеличивает скорость передачи информации и повышает уровень ее понимания, но и способствует развитию таких важных для специалиста любой отрасли качеств, как интуиция, образное мышление.

Новой для сферы обучения является интерактивность компьютерной графики, благодаря которой в процессе анализа изображений существует возможность динамически управлять их содержанием, формой, размерами и цветом, рассматривать графические объекты с разных сторон, приближать и удалять их, менять характеристики освещенности и проделывать другие подобные манипуляции, добиваясь наибольшей наглядности. Таким образом, это возможность не только для пассивного созерцания графических иллюстраций, но и для активного исследования характеристик графических моделей изучаемых объектов или процессов [22].

1.2 Анализ основных направлений визуализации данных

1.2.1 Визуальная аналитика

Визуальная аналитика (Visual Analytics) представляет собой междисциплинарный раздел знаний, основанный на синтезе и дальнейшем совместном развитии методов и подходов из уже сложившихся областей научных исследований, таких как научная визуализация (Scientific Visualization), визуализация информации и информационных процессов (Information Visualization), анализ многомерных данных (Data Analysis).

Однако данный синтез является не просто суммированием методов и подходов из различных дисциплин. Он имеет своей целью анализ огромного объема информации, а целью этого анализа является обеспечение поддержки принятия оперативного и точного решения человеком. Поэтому визуальная аналитика включает в себя и компьютерную графику, и изучение законов человеческого восприятия, и построение интерфейсов, обеспечивающих наиболее быстрое и оптимальное восприятие. Визуальную аналитику часто называют «наукой аналитического обоснования, усиленной с помощью интерфейсов интерактивной визуализации» [5].

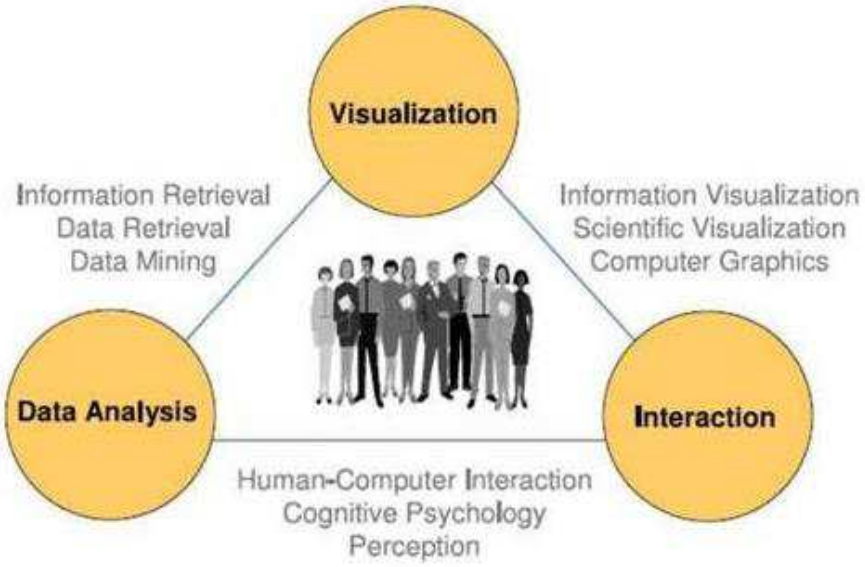

Рисунок 1 – Визуальная аналитика как результат междисциплинарного синтеза

Основные методы, подходы и алгоритмы визуальной аналитики описаны в работах [5-7]. В этих же работах приведен ряд примеров современного применения визуальной аналитики в различных сферах человеческой деятельности, а также приведены описания ряда программных продуктов, построенных на основе визуальной аналитики. Согласно [5-7], визуальная аналитика призвана организовать человеко-машинный интерфейс, усиливающий человеческие аналитические способности с помощью следующих методов:

- расширение оперативной памяти человека за счет использования визуализации;

- уменьшение области поиска путем представления большого объема данных в пространстве меньшей размерности;

- размещение информации в пространстве в соответствии с временными соотношениями;

- использование легких для восприятия представлений отношений между объектами;

- организация возможности восприятия большого числа потенциальных событий;

- организация управляемой среды для работы пользователя в пространстве параметрических значений;

- организация визуального представления и интерфейсов, обеспечивающих человеку возможность сразу видеть, исследовать и понимать огромные информационные объемы.

Реализация процедур визуальной аналитики применительно к большим информационным объемам подчиняется общей схеме (рисунок 3). Первым этапом в данной последовательности процедур является подготовка данных. Обычно считается, что рассматриваемые данные являются сложными и гетерогенными, имеют смешанный тип и получены из разных источников. Также считается, что рассматриваемые данные обладают разным уровнем качества. На предварительном этапе проводится фильтрация данных, удаление шумов и преобразование данных с целью совместной обработки. Источники данных (например, хорошо организованные базы данных) выстраиваются таким образом, что бы обеспечить непрерывный входной поток данных. Далее данные обрабатываются и переводятся в абстрактный вид с помощью математических моделей, статистических моделей и методов Data Analysis. Далее проводится реализация визуального представления полученных абстракций с целью выявления характерных черт, включая общности и аномалии. Это дает возможность пользователю провести визуальную оценку полученных результатов. Организация визуальных представлений должна быть оптимизирована с учетом всех возможностей и ограничений человеческого зрения и восприятия. Интерактивный режим визуализации позволяет пользователю проводить обработку данных и получать необходимую информацию, обеспечивающую возможность трактовки рассматриваемого информационного объема.

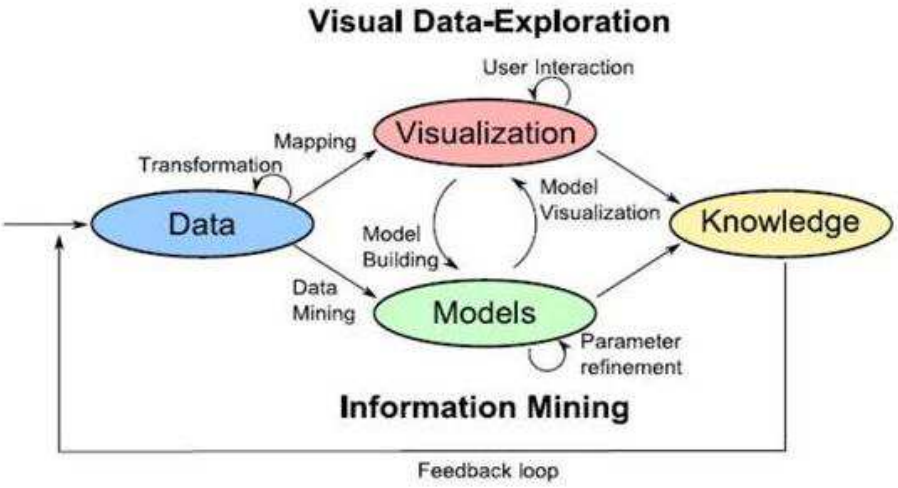

Рисунок 2 – Общая схема процесса реализации визуальной аналитики многомерных данных

При реализации этих процедур инструменты визуальной аналитики должны обеспечивать пользователю ряд возможностей, таких как:

– возможность визуализации данных с разных концептуальных точек зрения и на всех этапах: от «сырых» первичных данных до полученных абстракций;

– визуализация огромного количества информации в малом пространстве;

– извлечение из данных образов, отражающих общности, аномалии, соотношения между объектами и отдельные события;

– моделирование и верификация гипотез;

– оперативный обмен информацией.

1.2.2 Научная визуализация

Научная визуализация – это создание иллюстративной части для научной практики, представление данных эксперимента в визуальной форме. Она необходима ученым других областей, чтобы выявить закономерности, найти решение некоторой задачи и многое другое.

Научная визуализация обеспечивает этапы анализа и интерпретации результатов математического и компьютерного моделирования. Специализированные системы позволяют увидеть особенности моделируемых явлений и процессов в виде качественной и достоверной «картинки». Достоверность результатов визуализации может быть обеспечена в рамках исследований по верификации и валидации визуализации. Верификация визуализации подразумевает наличие некоторой формальной модели. Важной задачей является участие математика, проводящего исследование, в проектировании видов отображения для ускорения всего процесса разработки и улучшения качества работы системы визуализации [8].

Суть метода научной визуализации заключается в том, что исходным анализируемым данным ставится в соответствие некоторая их статическая или динамическая графическая интерпретация, которая визуально анализируется, а результаты анализа этой графической интерпретации (графических данных) истолковываются исследователем по отношению к исходным данным [2].

При такой трактовке научной визуализации как единое целое рассматриваются собственно визуализация анализируемых научных данных (что соответствует, как было отмечено выше, трактовке научной визуализации в широком смысле) и визуальный анализ ее результатов. Исходные данные, анализируемые методом научной визуализации, могут иметь различную природу. Наряду с этим, могут различаться и цели анализа исходных данных. Соответственно, могут различаться и используемые графические данные. Этот метод дает возможность использовать огромные потенциальные возможности пространственно-образного мышления исследователя в процессе анализа различных научных данных.

Развитие и усложнение этого метода, а также расширение класса задач, для решения которых он используется, делает актуальным дальнейшую разработку его научных и инженерных основ.

1.2.3 Визуализация информации и информационных процессов

Информационная визуализация или визуализация информации - это изучение (интерактивных) визуальных представлений абстрактных данных для усиления человеческого познания. Абстрактные данные включают как числовые, так и не численные данные, такие как текстовая и географическая информация.

Область визуализации информации возникла «от исследований в области взаимодействия человека и компьютера, информатики, графики, визуального дизайна, психологии и бизнес-методов. Она все чаще применяется в качестве важного компонента в научных исследованиях, цифровых библиотеках, интеллектуальном анализе данных, финансовых данных анализ, исследование рынка, контроль производственного производства и открытие лекарств» [9].

Визуализация информации предполагает, что «визуальные представления и методы взаимодействия используют в себе широкий путь пропускания человеческого глаза в сознание, чтобы пользователи могли видеть, исследовать и понимать большое количество информации одновременно. Информационная визуализация была сосредоточена на создании подходов к передаче абстрактных информации интуитивно понятными способами». [10]

Анализ данных является неотъемлемой частью всех прикладных исследований и решения проблем в промышленности. Наиболее фундаментальные подходы к анализу данных - это визуализация (гистограммы, диаграммы рассеяния, поверхностные графики, древовидные карты, параллельные координаты и т.д.), Статистика (тест гипотезы, регрессия, СПС и т.д.), Интеллектуальный анализ данных (объединение добычи и т.д.), и методы машинного обучения (кластеризация, классификация, деревья решений и т. д.). Среди этих подходов, визуализация

информации или анализ визуальных данных, наиболее полагаются на когнитивные навыки человеческих аналитиков и позволяют открывать неструктурированные действенные идеи, которые ограничены только человеческим воображением и творчеством. Аналитик не должен изучать какие-либо сложные методы, чтобы иметь возможность интерпретировать визуализацию данных. Информационная визуализация также представляет собой схему генерации гипотез, которая может быть и обычно сопровождается более аналитическим или формальным анализом, таким как статистическое тестирование гипотез.

1.3 Визуально интерактивное моделирование для представления 2D моделей сечений

Существуют различные виды визуализации информации, такие как: таблица или матрица, диаграмма связей, количественная диаграмма, различного рода графики, гистограммы, столбчатые диаграммы и т.д. Все эти методы отображают данные однородного характера, зачастую это изменение какого-либо параметра с течением времени. К сожалению, они не способны представить в удобном для человека виде данные более сложных структур. Для удобного анализа разнородных данных или же многомерных используют несколько другие способы: тепловые диаграммы, трехмерные графики, диаграммы, гистограммы (рисунки 2, 3).

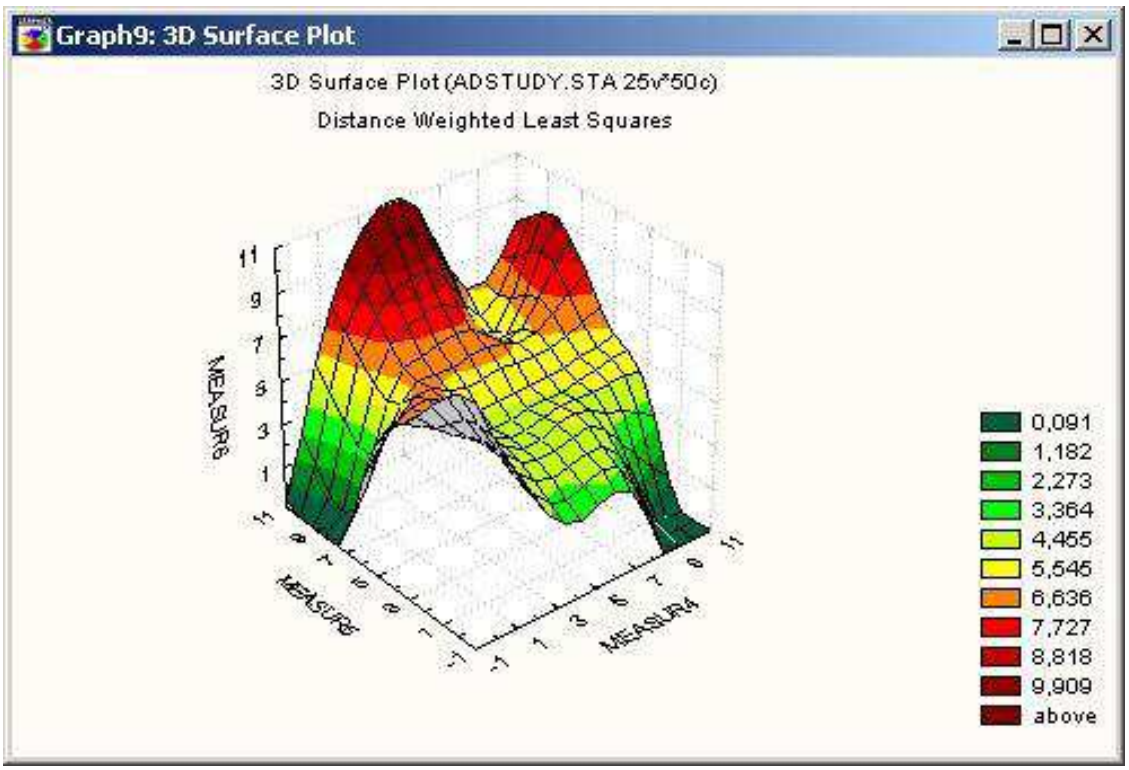

Рисунок 3 – Пример трехмерного графика

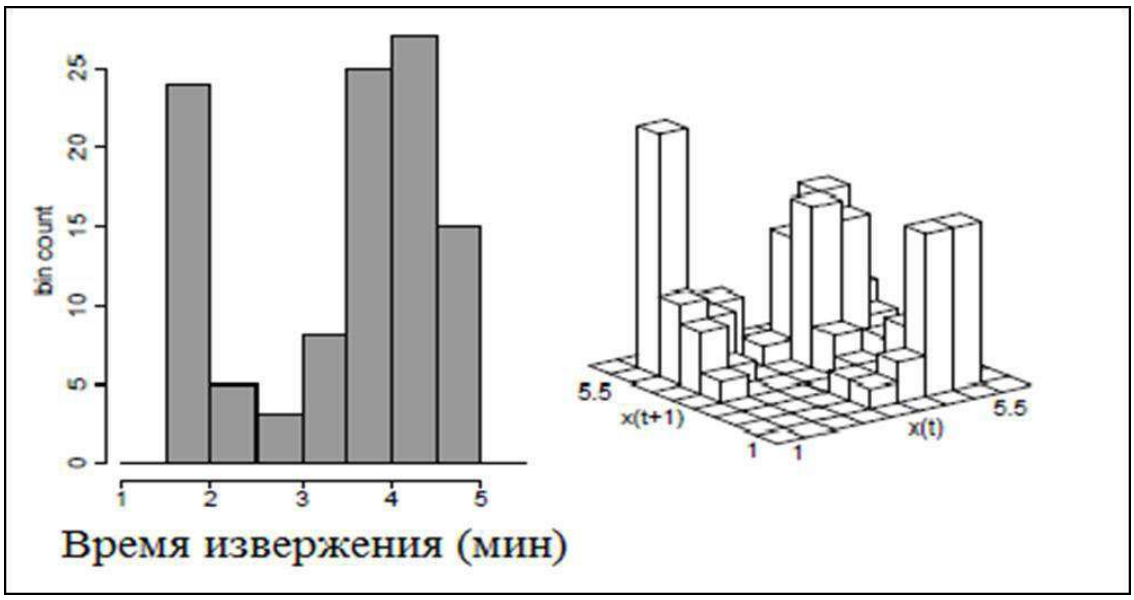

Рисунок 4 – Пример трехмерной диаграммы (справа)

Основная проблема подобных способов в том, что отображаемые поверхности перекрывают друг друга или вовсе не представляют возможным увидеть картину целиком. Тем самым снижается информативность и соответственно полезность отображаемых данных.

Необходимо понимать, что пространство, в котором существует человек, имеет трехмерный вид, то есть, каждую точку, лежащую в данном пространстве, возможно описать при помощи трех координат: x – длина, y – высота и z – длина. Проблема состоит в том, что человек не способен видеть окружающее пространство в трехмерном виде так, «как оно есть». Человеческий глаз имеет возможность только строить проекцию окружающего трехмерного мира на сетчатку, которую, в свою очередь, человек и воспринимает за 3D изображение объемного пространства вокруг себя. Другими словами, человек способен видеть только две координаты: x и y, третью координату z мозг рассчитывает самостоятельно, исходя из других параметров проекции видимого пространства, физических законов и самостоятельного опыта. Часто бывает так, что, выполняя различные практические задачи, человеку необходимо изучать какой-либо объект. Проблема в том, что исследуемый объект, как и пространство, в котором он находится, имеет трехмерную сущность. Следовательно, изучая, данный объемный объект, человеку необходимо знать, что происходит с объектом не только на видимом его участке, но и на скрытых от взора исследователя (сбоку, слева, внутри). Как пример подобной практической задачи, возможно представить получение и отображение данных, часто используемое в медицине и других областях человеческой деятельности, называемое томографией. Томография, простыми словами, представляет собой получение послойного изображения внутренней структуры изучаемого объекта. Пример подобного способа отображения данных представлен на рисунке 4 на так называемых томограммах – последовательно сделанных срезах исследуемого объекта (в данном случае, человеческого мозга).

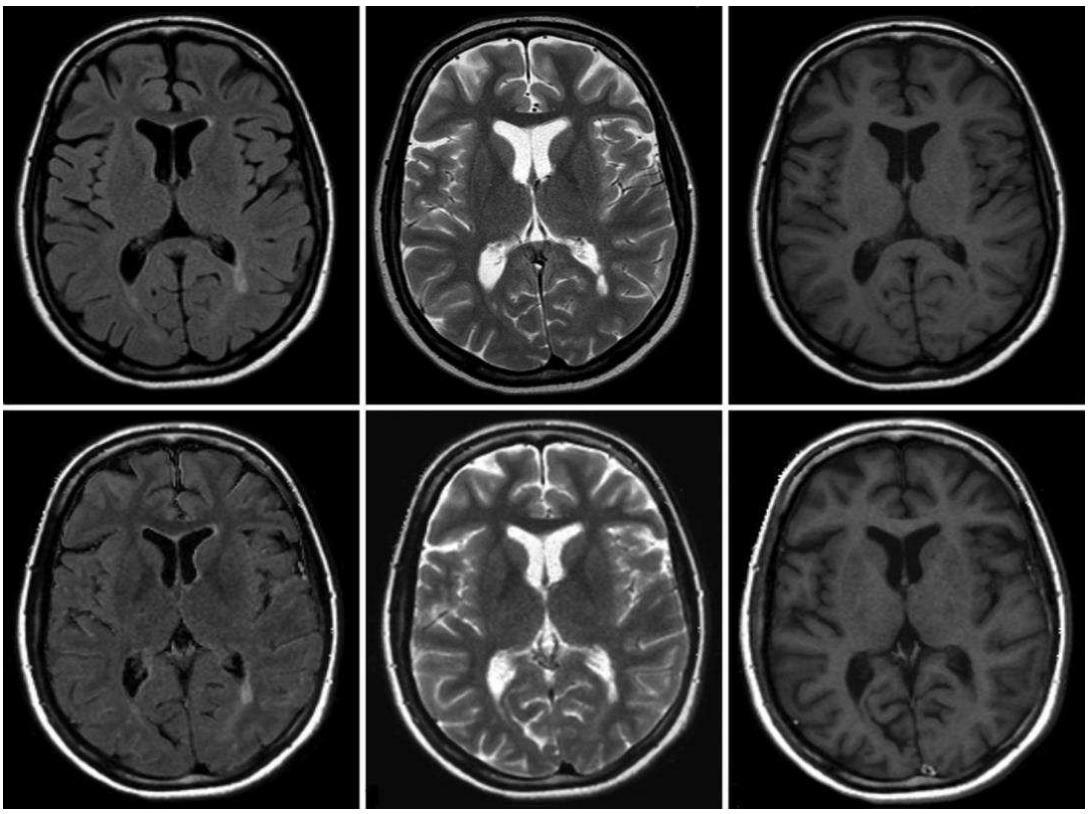

Рисунок 5 – Томограмма человеческого мозга

Используя данный вид получения и отображения данных, исследователь имеет возможность изучения внутренней составляющей объекта на статичных изображениях с помощью срезов, последовательно выполненных на одной из координатных осей через определенный интервал. Однако, необходимо понимать, что данный вид представления данных имеет ряд недостатков, главным их которых является отсутствие возможности динамичного отображения данных. Другими словами, из-за того, что данные представлены статично, нет возможности изучить объект с разных точек, увеличить интересующие участки или взглянуть на объект в целом, изменить ось среза и так далее.

Разрабатываемый подход визуального моделирования многомерных данных предназначен прежде всего для того, чтобы на практике проверить возможность представления многомерных данных и данных больших объемов таким образом, чтобы отображаемый объем данных был комфортен и пригоден для исследования в рамках используемой компьютерной модели.

Особенностью предложенного подхода визуально-интерактивного моделирования является динамическое представление 2D моделей сечений исследуемых данных. Эффект многомерности достигается за счет выбора 2D моделей и вариативности направлений динамических сечений.

Глава 2 Технологии стерео визуализации изображений

2.1 Методы представления объемного изображения

Стереоизображение – картина или видеоряд, использующий два отдельных изображения, позволяющих достичь стереоэффекта. Первый стереофотоаппарат был сконструирован и запатентован российским изобретателем Иваном Фёдоровичем Александровским в 1854 г.

Чтобы создать стереоизображение в программе трёхмерного моделирования, надо сделать двойной рендеринг сцены – с двух камер, соответствующих глазам наблюдателя.

Для создания и просмотра стереоизображения используются различные методы и устройства.

Метод параллельного взгляда позволяет посмотреть полноцветную стереокартинку безо всякого оборудования, стереоэффект достигается за счет сведения глаз дальше плоскости изображения. Используются два снимка, сделанные с разных ракурсов. Способ пригоден только для просмотра небольших изображений (шириной до 60–70 мм каждое), что обусловлено межзрачковым расстоянием человека.

Метод перекрёстного взгляда (cross-eye) аналогичен предыдущему, но глаза сводятся перед изображением. Преимущество – стереопара может быть любого размера (но при слишком большом угловом расстоянии между компонентами возникает сильное напряжение глаз); недостатки — мнимое изображение возникает между экраном и наблюдателем, что ограничивает размеры изображённого объекта либо превращает его в «кукольную копию».

Параллельную стереопару можно превратить в перекрёстную перестановкой составляющих.

Анаглиф-очки – разноцветные очки, вместо линз у которых вставлены светофильтры цветов CMY. Дешёвый, но довольно эффективный метод. Он не обеспечивает правильную передачу цвета, однако нервная система довольно хорошо интерпретирует его. Время адаптации — около 30 секунд, после длительного использования на пропорциональный период нарушается цветовосприятие.

Затворные стереоочки. На экран проецируется то картинка для левого глаза, то для правого. Соответственно, очки открывают обзор то левому глазу, то правому. Применяются в 3D-кино формата XpanD. Изредка используются в компьютерных играх, так как позволяют задействовать обычный ЭЛТ-монитор (но с мощной видеоплатой — нагрузка на неё повышается вдвое). ЖК-монитор годится не каждый — истинная частота обновления у большинства из них не превышает 30—75 Гц (имеется в виду фактическое время перестроения ЖК-цепочек, а не частота развёртки). Примером такой технологии является nVIDIA 3D Vision. Для использования 3D Vision нужен ЖК, плазменный или OLED-монитор с частотой развёртки 100 Hz или выше, видеокарта от nVIDIA с 3D Vision и специальные очки. Начиная с 2009—2010 годов в мире началось массовое производство телевизоров, работающих по этому принципу. В апреле 2010 года в России началось конвейерное производство 3D-телевизоров Samsung в Калужской области. Зритель надевает ЖК-очки, которые поочерёдно (с частотой 60 Гц) затемняют левый и правый глаза человека, телевизор при этом показывает 120 изображений в секунду.

Поляризационные стереоочки. Сами очки несколько дороже анаглифных и требуют прецизионного спецоборудования, вдобавок киноэкран должен быть металлизированным, чтобы не было деполяризации света. Однако (кроме понижения яркости и дороговизны) выраженных недостатков не имеют. Обычно применяются в стереокинотеатрах. Имея два

схожих проектора, экран и некоторое количество поляризационной плёнки от неисправного ЖК-монитора, можно самостоятельно воспроизвести в большей или меньшей степени такой стереоэффект.

Основанные на линейной поляризации (дешевле, но при наклонах головы стереоэффект теряется). Применяется в 3D-кино формата IMAX 3D.

Основанные на круговой поляризации (дороже). Применяется в 3D-кино формата RealD Cinema. На чересстрочной круговой поляризации работают 3D-телевизоры LG Electronics. Существуют очки и с одинаковыми фильтрами на обоих глазах, они используются, чтобы наладить многопользовательскую игру или просмотр одновременно двух передач на одном телевизоре.

Стереоочки с многополосными фильтрами — обеспечивают стереоэффект за счёт того, что линзы пропускают лишь узкие полосы красного, зелёного и синего. Проекционное оборудование относительно дёшево, но сами стереоочки дороги. Применяется в 3D-кино формата Dolby 3D.

Стереоскоп — оптический прибор с двумя окулярами; обычно используется для просмотра стереослайдов, но не составляет сложности вложить туда КПК или коммуникатор с продолговатым экраном высокого разрешения (например, Nokia E90).

Стереодисплей — оптический инструмент, с помощью которого два плоскостных изображения комбинируются таким образом, что наблюдатель получает впечатление рельефного предмета.

Виртуальный шлем (VR HMD) — шлем, который показывает для каждого глаза отдельные изображения. В результате чего получается стереоэффект.

2.2 Стереоизображение

Особенностью стереоизображения является то, что оно состоит из повторяющихся отрезков. Период повторения и его структура показана на рисунке 6.

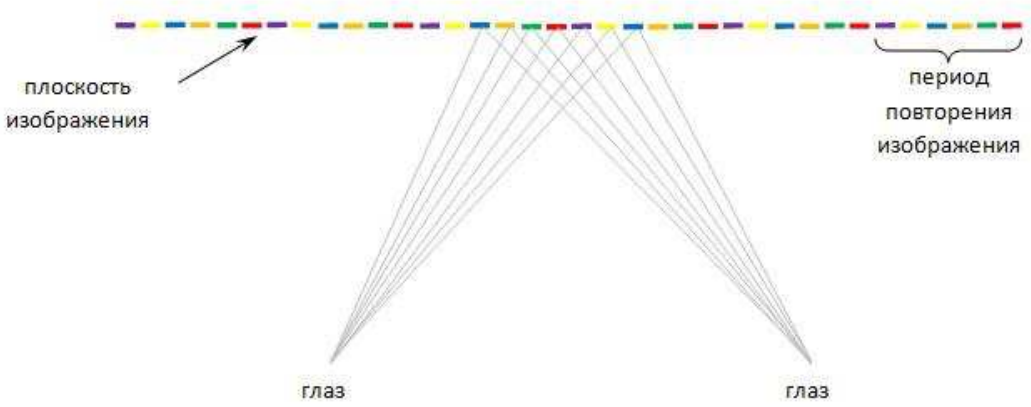

Рисунок 6 – Повторяющиеся фрагменты стереоизображения

Ширина данного повторяющегося отрезка не должна превышать расстояние между глаз человека. Она должна находиться в рамках 1/3 и 2/3 данного расстояния. От ширины зависит, насколько объемное изображение мы сможем получить, и как сильно будут уставать глаза при просмотре изображения подобного формата.

Для того чтобы изображение стало объемным, глаза человека должны быть сфокусированы за плоскостью изображения. Это показано на рисунке 7. При условии что изображения, как для левого глаза, так и для правого будут совпадать. В случае, когда изображение полностью состоит из повторяющихся отрезков, мнимое изображение будет точно таким же, как и само изображение. При этом будет казаться, что расположение изображения стало дальше.

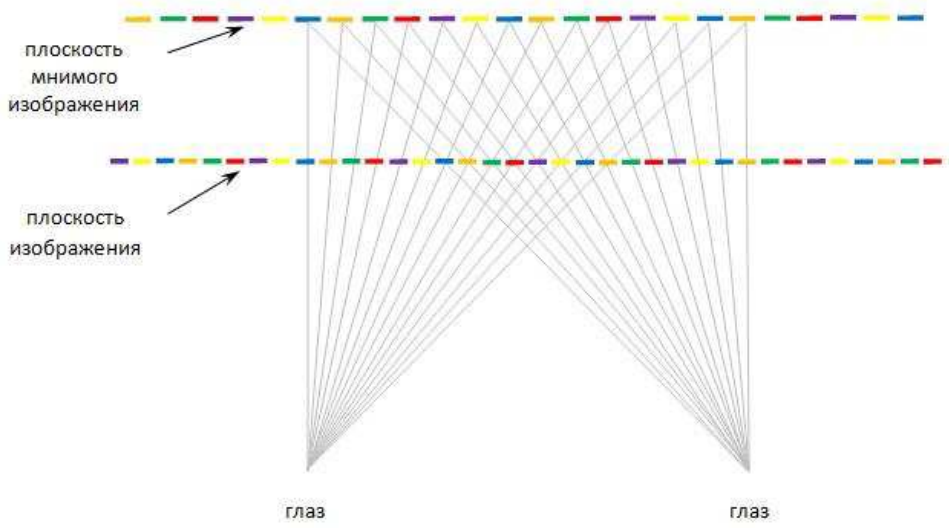

Рисунок 7 – Правильная фокусировка глаз

Эффект объемного изображения достигается путем сдвига различных отрезков изображения. Вследствие чего некоторые фрагменты изображения кажутся ближе к нам, некоторые дальше.

Рисунок 8 показывает, как при сдвиге нескольких элементов реального изображения изменяется структура мнимого изображения.

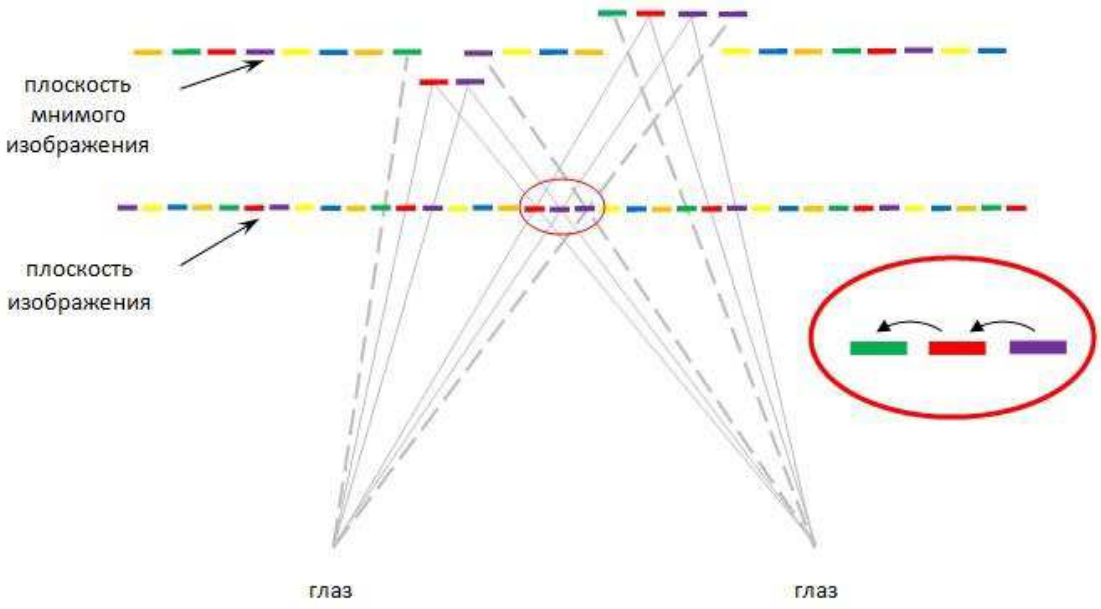

Рисунок 8 – Изменение изображения после сдвига элементов

При изменении изображения образуются артефакты, равноудаленные от основной плоскости мнимого изображения. Артефакты, расположенные перед основной плоскостью изображения кажутся ближе, чем артефакты, расположенные позади плоскости. Пунктирной линией отмечены элементы, которые видны лишь одному глазу, наш мозг сам додумывает место их расположения.

Стоит заметить, что сдвиг должен быть меньше половины ширины повторяющегося отрезка. В тех случаях, когда величина сдвига слишком близка к ширине отрезка, возникнут сложности с определением рельефа данной плоскости.

Для того, чтобы отображать объекты только перед плоскостью мнимого изображения, необходимо бороться с артефактами, находящимися позади плоскости. Это можно сделать с помощью сдвига нужных элементов в каждом из периодов. Подобный сдвиг изображен на рисунке 9.

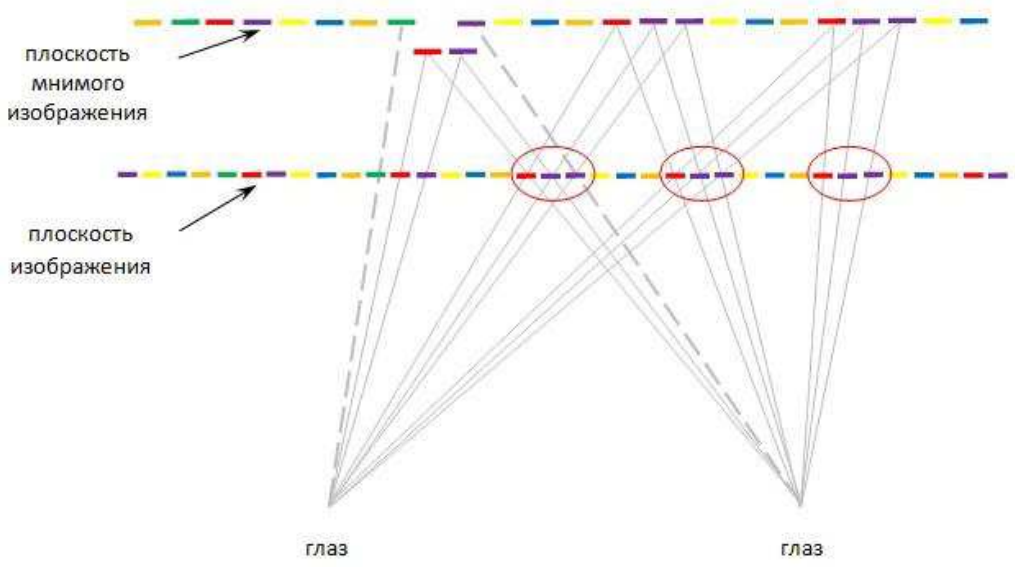

Рисунок 9 – Изображение объектов только перед мнимой плоскостью

Если осуществить сдвиг больше чем на одну позицию, то выпуклая область будет казаться нам еще ближе. Наглядно это можно увидеть если сравнить два рисунка: 9 и 10.

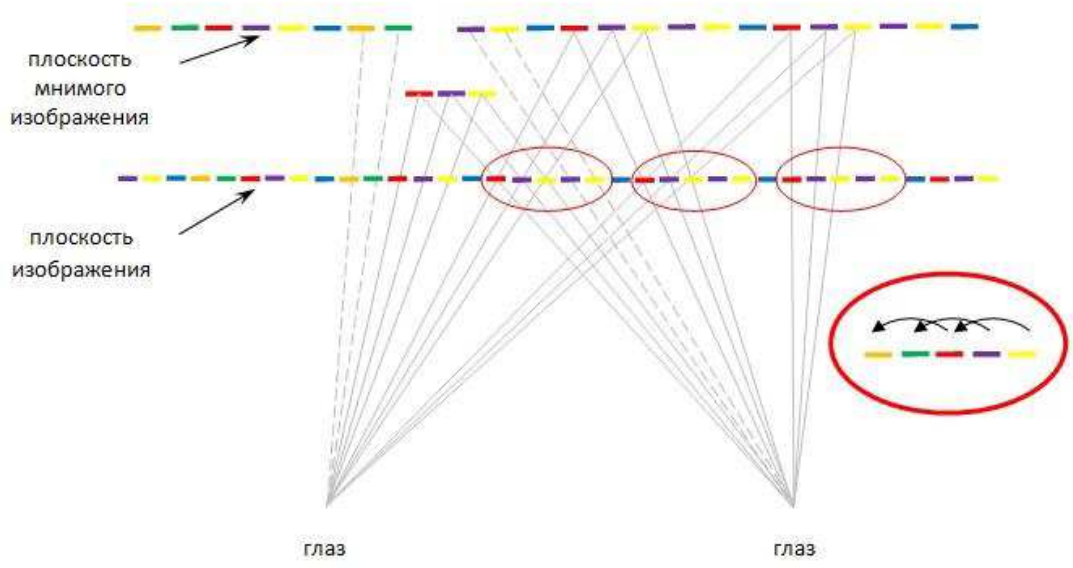

Рисунок 10 – Сдвиг трех элементов на две позиции

Глава 3 Разработка программного модуля 3D визуализации данных

3.1 Цели и задачи разработки

Для тестирования методов визуализации была выбрана программная система Autodesk 3ds Max. Это полнофункциональная профессиональная программная система для создания редактирования трёхмерной графики и анимации, разработанная компанией Autodesk.

В качестве объекта для визуализации были выбраны полые сферы разных радиусов, в количестве от трех до шести штук, так что одна находилась в другой, напоминая упрощенную модель Земли. Слои имеют разные цвета или вовсе полностью прозрачны в зависимости от способа визуализации.

Цель разработки программного модуля реализовать один из трех способа визуализации данных:

- одиночный движущийся срез, объект имеет разную прозрачность слоев, метод изображен на речонке 6;

- пять линий контура в движении, линии имеют разную яркость, сам объект полностью прозрачный, метод изображен на рисунке 7;

- туннельный способ, слои наслаиваются друг на друга с течением времени от центра объекта, метод изображен на рисунке 8).

Задачи тестирования методов визуализации:

– исследовать имеющиеся методы визуализации;

– предложить собственные методы;

– протестировать предложенные методы;

– выявить наиболее перспективный метод для дальнейшей разработки.

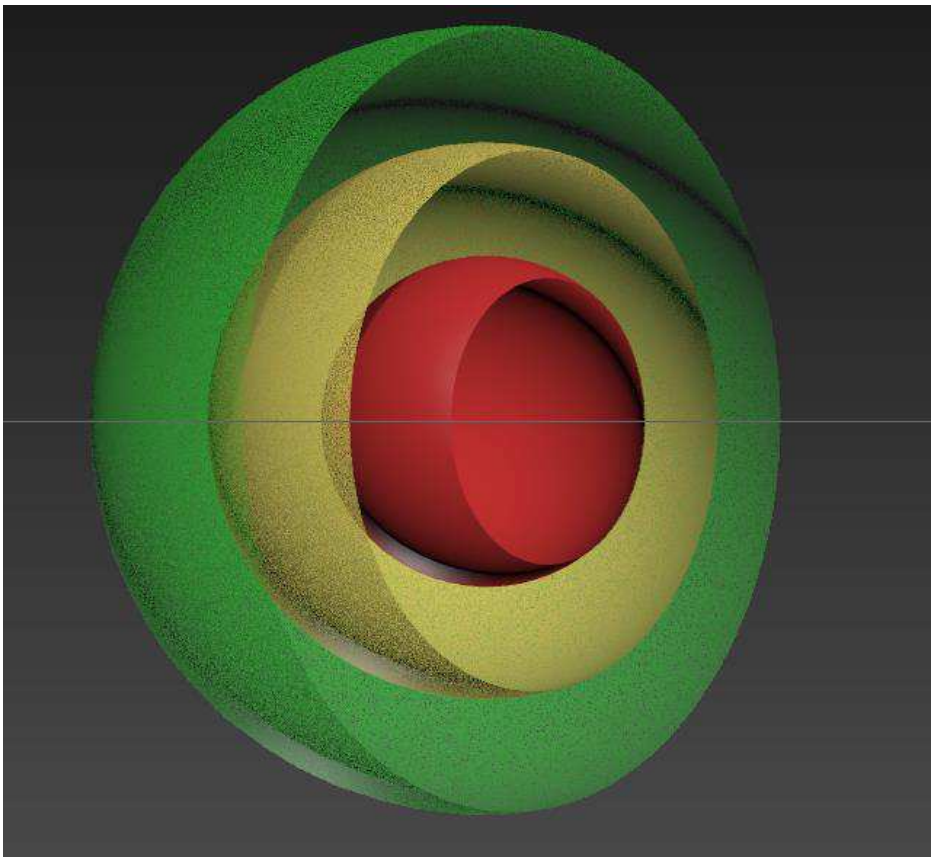

Рисунок 11 – Одиночный срез

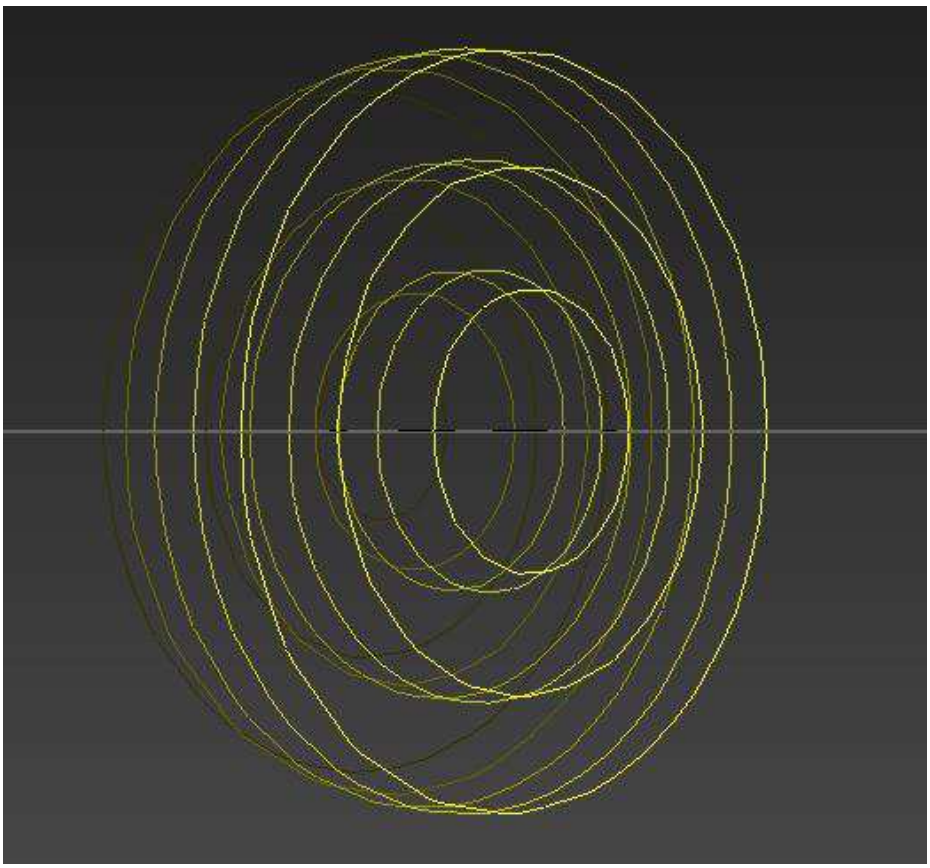

Рисунок 12 – Линии контура

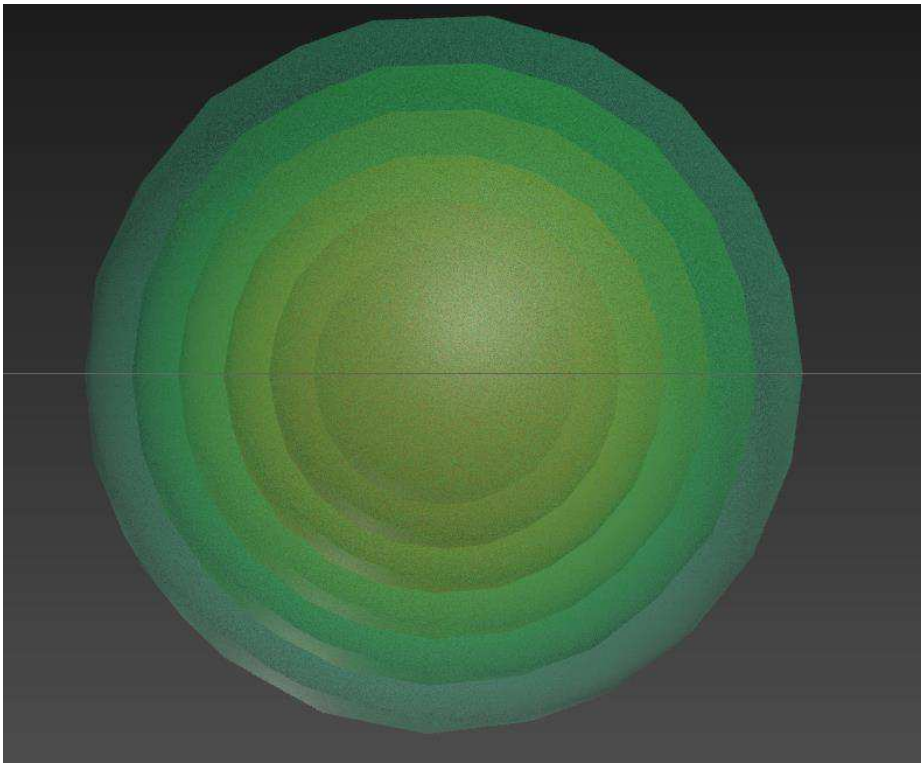

Рисунок 13 – Туннельная модель

Таким образом добавление срезов, прозрачности и динамики позволяют увидеть структуру и природу построения объекта целиком, поэтому данный метод является наиболее перспективным в дальнейшей разработке. Так же важную роль играет изменение направления движения среза и точки обзора.

3.2 Описание алгоритма

OpenGL (англ. Open Graphics Library) – открытая графическая библиотека OpenGL, которая является одним из самых популярных прикладных программных интерфейсов API (Application Programming Interface) для разработки приложений в области двумерной и трехмерной графики [28].

Для стабильной работы OpenGL необходимо выполнить первоначальную настройку параметров, таких как: переход между кадрами, тип буферизации, сжатие и фильтрацию. Далее необходимо установить точку обзора (камеру) и создать рабочее окно, в котором будет отображаться графика OpenGL. На этом шаге настраивается угол обзора камеры, её позиция и размеры окна. Третьим шагом устанавливаются переменные, необходимые для рисования окружностей: их радиусы и начальная позиция. Когда все подготовки закончены, запускается рабочее окно программы, следом отрисовывается его содержимое. После этого перерисовка фигур в окне происходит с определенным интервалом, что создает эффект анимации. Программа продолжит перерисовывать кадр до тех пор, пока окно программы не будет закрыто, в этом случае OpenGL прекратит свою работу.

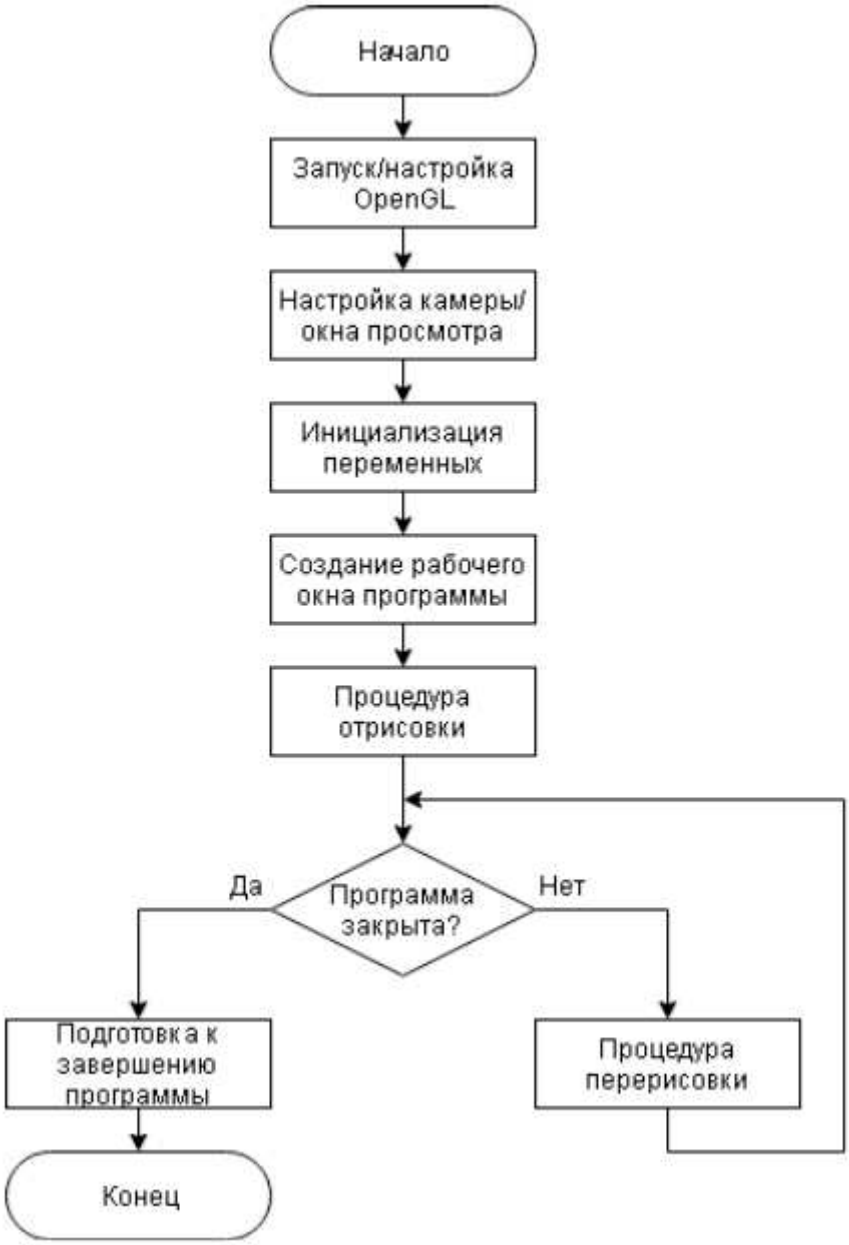

Рисунок 14 – алгоритм работы программы.

3.3 Описание программного кода

OpenGL является средой для разработки интерактивных 2D и 3D графических приложений. OpenGL позволяет быстро разрабатывать приложения, включающие в себя рендеринг, текстурирование, спецэффекты и другие мощные функции визуализации [5].

GLUT – библиотека для использования OpenGL, реализующий простой оконный интерфейс API [4].

Необходимы выбрать среду разработки для работы с графикой, PascalABC.NET хорошо подходит для этой задачи, язык программирования PascalABC содержит в себе библиотеки OpenGL и GLUT. А среда разработки является бесплатной, простой и достаточно мощной для решения подобных задач [3].

Программный файл начинается с подключения динамических библиотек, для работы с формой, при помощи которой осуществляется вывод трехмерной графики. Фрагмент этого программного кода изображен на рисунке 15.

```
{$reference System.Windows.Forms.dll}
{$reference System.Drawing.dll}
```

Рисунок 15 – Подключение динамических библиотек

Далее подключаются необходимые модули из библиотек, модули system направлены на работу с формой (создание и обработка событий), так же подключаем OpenGL для работы с графикой и создаем новый класс, который наследуется от стандартного класса. Фрагмент этого программного кода изображен на рисунке 16.

```
uses System, System.Drawing, System.Windows.Forms, OpenGL;

type Form1 = class (Form)
```

Рисунок 16 – Создание класса

После создания класса необходимо определить переменные: _hdc указывает на саму форму, timer и components служат вспомогательными

переменными для создания таймера. Фрагмент этого программного кода изображен на рисунке 17.

```
private
 _hdc : HDC;
 timer: Timer;
 components: System.ComponentModel.IContainer;
```

Рисунок 17 – Определение перменных

Переменные для работы с окружностями: R1, R2, R3. R_m – радиус текущей окружности. X, Y, Z – точки на соответствующих осях координатной плоскости. Fi – угол от центра окружности. Фрагмент этого программного кода изображен на рисунке 18.

```
R1   : double;
R2   : double;
R3   : double;
R_m  : double;
Z    : double;
X    : double;
Y    : double;
Fi   : real;
```

Рисунок 18 – Переменные для работы с окружностями

Для того чтобы использовать OpenGL, его необходимо настроить: установка буфера глубины (glEnable), его размер (glDepthFunc), указать матрицу перспективы (glHint), то как мы будем видеть трехмерные объекты на двухмерно поверхности монитора. Для удобства настроек все значения взяты по умолчанию. Фрагмент этого программного кода изображен на рисунке 19.

```
procedure Init;
begin
  // Настройка OpenGL
  glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LEQUAL);
  glClearDepth(1.0);
  glHint(GL_PERSPECTIVE_CORRECTION_HINT,GL_NICEST);
```

Рисунок 19 – Настройка OpenGL

После этого необходимо выполнить настройку окна просмотра. Местоположение рабочего окна на экране монитора, его высота и ширина, а также перспектива. Фрагмент этого программного кода изображен на рисунке 20.

```
// Настройка камеры/окна просмотра
glMatrixMode( GL_PROJECTION );
glLoadIdentity;
glViewport(0, 0, Width, Height);
gluPerspective(45.0, Width/Height, 0.1, 1000.0);
```

Рисунок 20 – Настройка окна просмотра

Для некоторых из переменных необходимо задать начальные значения, радиусы окружностей и смещение по оси Z (точка начала анимации). Фрагмент этого программного кода изображен на рисунке 21.

```
// Инициализация переменных
R1 :=  0.5;
R2 :=  0.75;
R3 :=  2;
Z  := -2;
```

Рисунок 21 – Инициализация переменных

Настройки выполнены, переходим к рисованию, функция Render(). Она будет вызываться каждые 10млс (миллисекунд) и рисовать все то, что будет в ней описано. Сначала происходит заливка фона в светло-серый. Фрагмент этого программного кода изображен на рисунке 22.

```
procedure Render(sender: Object; e: EventArgs);
var
  i    : integer;

begin
  // Заливка фона
  glClearColor(single(0.75), single(0.75), single(0.75), single(0.0));
  glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT);
```

Рисунок 22 – Функция Render

Настройка камеры обзора так же производится внутри этой функции, поскольку точка обзора может изменятся с течением времени. Задаем точку местонахождения камеры и точку, на которую будет направлена камера. Фрагмент этого программного кода изображен на рисунке 23.

```
// Установка наблюдателя/камеры
glLoadIdentity();
gluLookAt(-7.0,0.0,7.0,  0.0,0.0,0.0,  0.0,1.0,0.0);
```

Рисунок 23 – Расположение камеры

Функции необходимые нам для рисования объектов: glColor3f() – цвет кисти, тот цвет в который будут окрашены последующие фигуры, glBegin()/glEnd() – соединяет линией точки указанные между операторами gl, так же соединяет первую с последней создавая замкнутую фигуру, glVertex3f() – задает точку в пространстве. Для начала необходимо нарисовать оси x, y, z, для понимания где именно в пространстве находится объект, каждая пара операторов Begin/End отрисовывает отдельную ось. Фрагмент этого программного кода изображен на рисунке 24.

```
// Рисуем объект
glColor3f(0.125, 0.125, 0.125);
glBegin(GL_LINES);
  glVertex3f(-2, -2, -2);
  glVertex3f(2, -2, -2);
glEnd();
glBegin(GL_LINES);
  glVertex3f(2, -2, -2);
  glVertex3f(2, 2, -2);
glEnd();
glBegin(GL_LINES);
  glVertex3f(2, -2, -2);
  glVertex3f(2, -2, 2);
glEnd();
```

Рисунок 25 – Оси X, Y, Z

Теперь необходимо обозначить квадрат, который и будет нашим срезом (и отличаться по цвету от осей). Срез двигается по оси Z, от -2 до 2, это обусловлено наибольшим из радиусов, который равен двум, поэтому рисовать квадрат в случае если Z>2 нет смысла. Так же местоположение квадрата должно зависеть от переменной Z, следовательно третьим параметром функции glVertex3f() будет переменная Z. Фрагмент этого программного кода изображен на рисунке 25.

```
glColor3f(0.0, 0.7, 0.7);
if Z <= 2 then
begin
  glBegin(GL_LINE_LOOP); //квадрат
    glVertex3f(2, 2, Z);
    glVertex3f(2, -2, Z);
    glVertex3f(-2, -2, Z);
    glVertex3f(-2, 2, Z);
  glEnd;
end;
```

Рисунок 25 – Отрисовка квадрата

Далее рисуем окружности. Для каждой из них необходимо задать разный цвет: красный, желтый и зеленый. Вычислить радиус текущей окружности для положения среза по оси Z и нарисовать соответствующую окружность, состоящую из точек. Текущий радиус рассчитывается по формуле:

$$R\_m = \sqrt{R_i^2 - Z^2}, \quad (1)$$

где R_i – радиусы окружностей, Z – текущее положение среза на оси z. Переменная Fi обозначает на какой угол необходимо отклониться для того, чтобы нарисовать точку, также с её помощью вычисляется положение точек на осях x и y по формулам:

$$X = R\_m * \sin(Fi), \quad (2)$$

для точки на оси x, и для точки на оси y:

$$Y = R\_m * \cos(Fi). \quad (3)$$

Далее будет следовать следующая окружность, такого же цвета, но более прозрачного, для эффекта градиента. Координата Z, для каждой последующей окружности будет уменьшатся на 0.01, для того что бы окружности находились вплотную друг к другу, для начала нарисуем красную. Фрагмент этого программного кода изображен на рисунке 26.

```
glColor3f(1, 0.0, 0.0);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_1
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z);
  end;
glEnd;

glColor3f(0.95, 0.15, 0.15);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1.01));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_2
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z-0.01);
  end;
glEnd;
```

Рисунок 26 – Отрисовка окружностей

Таких окружностей 5, с плавным затуханием. Полностью показывать код программы смысла нет, т.к. он однообразный и занимает много места. Стоит обратить внимание на желтый круг, который мы рисуем после красного. Фрагмент этого программного кода изображен на рисунке 27.

```
glColor3f(1, 1, 0.0);
R_m := Sqrt(Sqr(R2) - Sqr(Z+0.5));
glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_1
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z);
  end;
glEnd;
```

Рисунок 27 – Отрисовка окружности

Поменялся цвет и переменная для расчета R_m, с R1 на R2. Зеленый круг по аналогии.

В конце нарисованного кадра необходимо сдвинуть координату Z на 0.01, для того, чтобы на следующем кадре окружности имели новые координаты для отрисовки. В случае если Z>2, этой переменной необходимо заново присвоить значение -2, таким образом анимация зациклится. Осталось только вывести отрисованный кадр на экран. Фрагмент этого программного кода изображен на рисунке 28.

```
// Делаем шаг по оси Z
Z += 0.01;
if Z >= 2.05 then Z := -2;
// Выводим содержимое на форму
glFlush();
SwapBuffers(_hdc);
```

Рисунок 28 – Зацикливание анимации

Так же необходимо иметь обработчики событий, таких как: изменение размера окна, которое повлечёт за собой изменение матрицы перспектив и объекты на экране будут обряжаться некорректно. Функция ResizeCL использует похожий код что и в начале программы при создании окна,

только с измененными высотой и шириной окна. Фрагмент этого программного кода изображен на рисунке 29.

```
procedure ResizeCl;
begin
  glMatrixMode( GL_PROJECTION );
  glLoadIdentity;
  glViewport(0, 0, Width, Height);
  gluPerspective(45.0, Width/Height, 0.1, 1000.0);
  glMatrixMode( GL_MODELVIEW );
  glLoadIdentity;
end;
```

Рисунок 29 – Обработчик событий

При создании объекта класса вызывается конструктор этого класса для нового объекта. Он служит для выполнения предварительных настроек объекта. именно в нем будет удобнее всего запустить таймер для отрисовки, настроить его, также запустить процедуру рисования окна , закрытие окна, изменении его размеров и применение заданных нами параметров для OpenGL. Шаг 7, блок-схемы алгоритма работы программы, является одним из обработчиков событий (Form_Closed) и вызывается в случае закрытия окна. Фрагменты этого программного кода изображены на рисунках 30 и 31.

```
constructor Create;
begin
  // Берем указатель на окно
  _hdc := GetDC(self.Handle.ToInt32());
  // Создаем таймер
  components := new System.ComponentModel.Container;
  timer := new System.Windows.Forms.Timer(self.components);
  timer.Enabled := true;
  timer.Interval := 10;
  // Инициализируем OpenGL
  OpenGLInit(self.Handle);
  // Настройка OpenGL
  Init;
  // Вызов каждую 1миллисек проц. Render
  timer.Tick += Render;
end;
```

Рисунок 30 – Создание таймера

```
// Процедура рисования окна
protected procedure OnPaint(e: System.Windows.Forms.PaintEventArgs); override;
begin
 //repeat
   Render(self, e);
 //until(false);
end;

// Процедура закрытия окна
procedure Form_Closed(sender : object; e : EventArgs);
begin
  // Уничтожения OpenGL
  OpenGLUninit(self.Handle);
end;

// Процедура выз. при изменения размера окна
procedure Form_Resize(sender: object; e : EventArgs);
begin
  ResizeCl;
end;
```

Рисунок 30 – События

В начале работы тела программы создается новый экземпляр класса. Далее вызываем конструктор, привязываем наши функции, задаем высоту и ширину окна. После чего вызываем процедуру Application.Run(), параметром которой будет являться объект нашего класса (форма). Фрагмент этого программного кода изображен на рисунке 32.

```
var f : Form1;
begin
  f := new Form1();
  f.Resize += f.Form_Resize;
  f.Closed += f.Form_Closed;
  f.KeyDown += KeyEventHandler(f.Form_KeyDown);

  f.Width  := 640;
  f.Height := 720;

  // FullScreen
  if (false) then begin
    f.TopMost := true;
    f.FormBorderStyle := System.Windows.Forms.FormBorderStyle.None;
    f.WindowState := System.Windows.Forms.FormWindowState.Maximized;
  end;

  // Запуск приложения
  Application.Run(f);
```

Рисунок 32 – Запуск программы

Полная версия программного модуля представлена в приложении А.

3.4 Описание результатов тестирования

Результатом работы программы является анимация среза по трем окружностям разных цветов и размером, показано на рисунках 15, 16, 17.

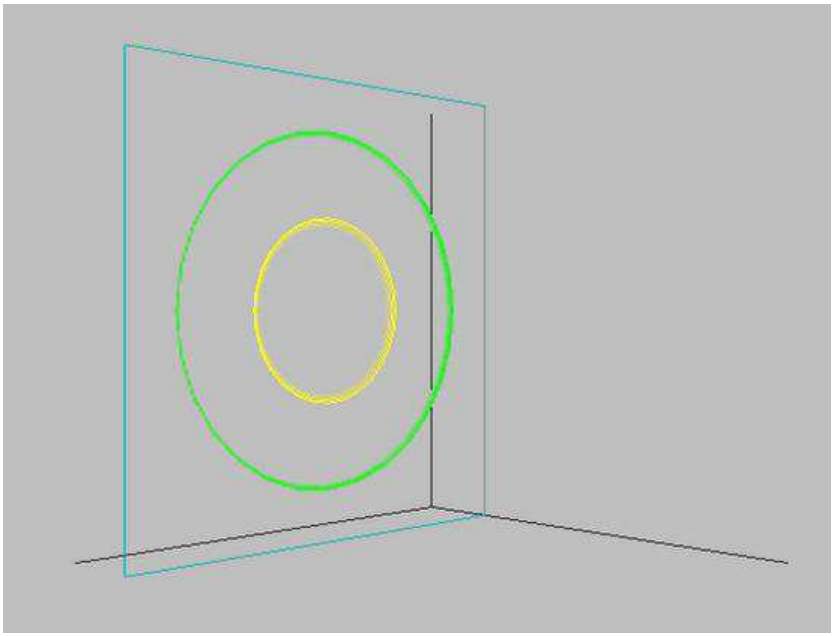

Рисунок 15 – Результат работы программы

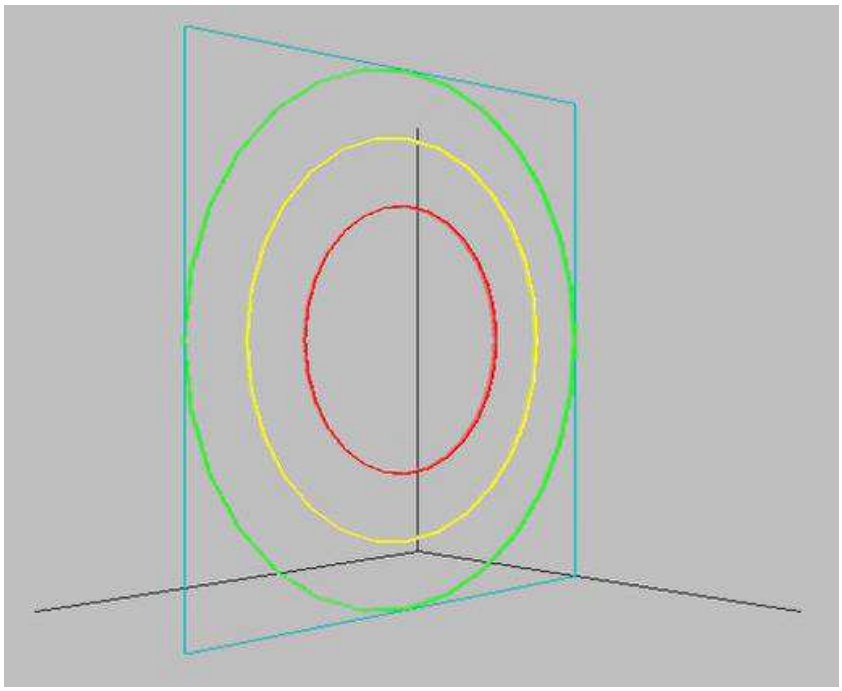

Рисунок 16 – Результат работы программы

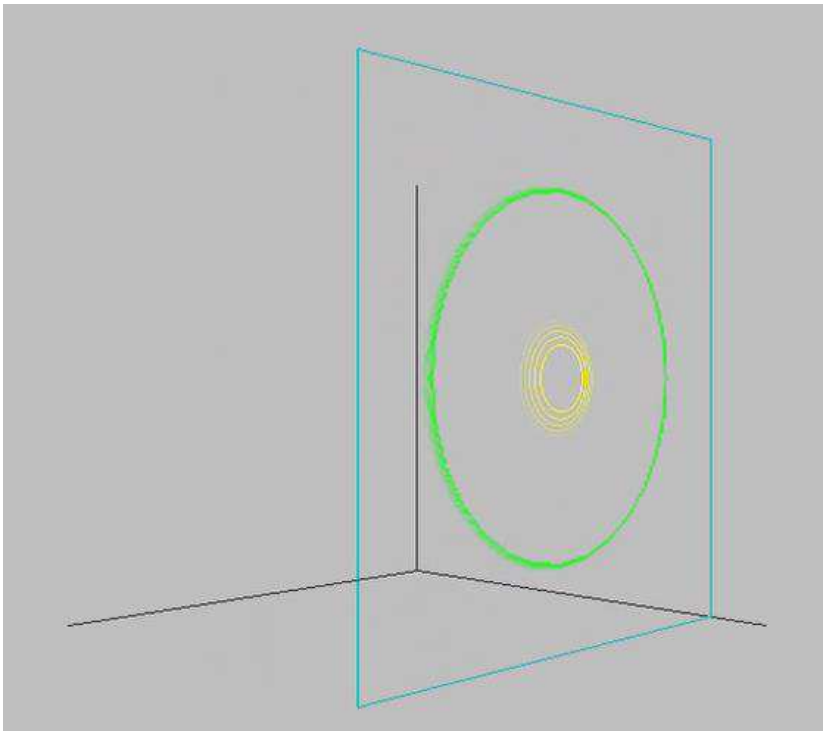

Рисунок 17 – Результат работы программы

Полученную анимацию необходимо было адаптировать к стерео формату для возможности просмотра на смартфоне с использование очков виртуальной реальности. Для этого необходимо: загрузить видео на смартфон и запустить его с помощью приложения для VR (Virtual Reality). Данное приложение имеет возможность конвертировать видео в стерео формат. Результаты изображены на рисунке 12.

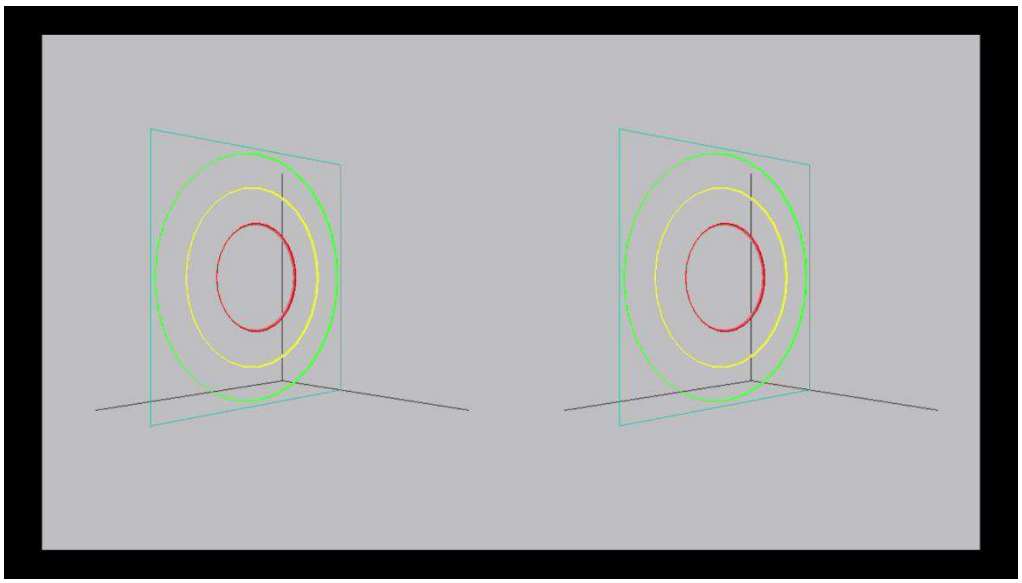

Рисунок 18 – Стерео формат

ЗАКЛЮЧЕНИЕ

В ходе выполнения магистерской работы были выявлены основные проблемы и исследована актуальность данной темы, представление и моделирование многомерных данных, позволяющее осуществлять визуальный анализ сложно структурированных объектов, осуществлен поиск научной информации, проведен анализ основных направлений и методов визуализации многомерных данных, рассмотрен подход визуально интерактивного моделирования представления 2D моделей сечений данных, предложены три метода визуализации данных и протестированы в среде Autodesk 3ds Max. в результате тестирования был выявлен наиболее перспективный метод. Данный метод был реализован в виде программного модуля с помощью языка программирования PascalABC.

Список использованной литературы

1. Айвазян С.А., Енюков И.С., Мешалкин Л.Д. Прикладная статистика. Классификация и снижение размерности. М.: Финансы и статистика, 1989.
2. Пилюгин В.В., Маликова Е.Е., Пасько А.А., Аджиев В.Д. Научная визуализация как метод анализа научных данных / Научная визуализация. Т.4, № 4, с.8-25, 2012, URL: http://sv-journal.org/2012-4/062139.html
3. Бондарев А.Е., Галактионов В.А. Анализ многомерных данных в задачах многопараметрической оптимизации с применением методов визуализации / Научная визуализация. Т.4, № 2, с.1-13, 2012, URL: http://sv-journal.org/2012-2/012139.html
4. Основы научной визуализации [сайт]. URL: http://ifes.mephi.ru/unl/ (дата обращения: 10.05.2014)
5. Thomas J., Cook K. Cook, Illuminating the Path: Research and Development Agenda for Visual Analytics. IEEE-Press, 2005.
6. Keim D. A, Mansmann F, Schneidewind J, Thomas J, Ziegler H: Visual analytics: Scope and challenges, Visual Data Mining: 2008, S. 82.
7. Keim D., Kohlhammer J., Ellis G. and Mansmann F. (Eds.), Mastering the Information Age – Solving Problems with Visual Analytics, Eurographics Association, 2010.
8. Авербух В.Л., Бахтерев М.О., Васёв П.А., Манаков Д.В., Стародубцев И.С. Развитие подходов к разработке специализированных систем компьютерной визуализации // Специализированные системы компьютерной визуализации – Екатеринбург.
9. Benjamin B. Bederson and Ben Shneiderman (2003). The Craft of Information Visualization: Readings and Reflections, Morgan Kaufmann ISBN 1-55860-915-6.

10. James J. Thomas and Kristin A. Cook (Ed.) (2005). Illuminating the Path: The R&D Agenda for Visual Analytics Archived 2008-09-29 at the Wayback Machine.. National Visualization and Analytics Center. p.30

11. Gorban A., Kegl B., Wunsch D., Zinovyev A. (Eds.), Principal Manifolds for Data Visualisation and Dimension Reduction, LNCSE 58, Springer, Berlin – Heidelberg – New York, 2007.

12. Зиновьев А. Ю. Визуализация многомерных данных, Красноярск, Изд. КГТУ, 2000. – 180 с.

13. Добронец Б. С., Попова О. А. – Визуально-интерактивное моделирование в условиях многомерности

14. Национальная библиотека им. Н. Э. Баумана [Электронный ресурс]: Режим доступа: https://ru.bmstu.wiki/OpenGL_(Open_Graphics_Library)

15. Современное программирование на языке Pascal [Электронный ресурс]: Режим доступа: http://pascalabc.net/

16. Lazarus and Free Pascal Wiki [Электронный ресурс]: Режим доступа: http://wiki.lazarus.freepascal.org/OpenGL_Tutorial/ru

17. OpenGL по шагам [Электронный ресурс]: Режим доступа: http://www.firststeps.ru/mfc/opengl/r.php?36

18. Численный вероятностный анализ неопределённых данных: монография / Б. С. Добронец, О. А. Попова; М-во образования и науки Российской Федерации, Сибирский федеральный ун-т, [Ин-т космических и информ. технологий]. - Красноярск : СФУ, 2014. - 166 с.

19. Добронец Б.С., Попова О.А. Численный вероятностный анализ неопределённых данных Сибирский федеральный университет, Институт космический и информационных технологий. Красноярск, 2014.

20. Добронец Б.С., Попова О.А. Элементы численного вероятностного анализа // Вест. Сиб. гос. аэрокосм. ун. им. акад. М.Ф. Решетнева. 2012. № 2. С. 19–23.

21. Пархоменко Д. Б. Применение многомерной визуализации для решения задач моделирования в робототехнике.

22. Першина, Е. Л., Попова, О. А., Чуканов, С. Н., Интеллектуальные системы поддержки принятия решений: комплексы программ, модели, методы, приложения // Федеральное агентство по образованию, ГОУ ВПО «Сибирская государственная автомобильно-дорожная акад. (СибАДИ)». Омск, 2010.

23. Добронец, Б. С., Попова, О. А., Представление и обработка неопределенности на основе гистограммных функций распределения и P-boxes // Информатизация и связь. 2014. № 2. С. 23–26.

24. Бондарев, А. Е., Галактионов, В. А., Шапиро, Л. З., Анализ многомерных данных в задачах многопараметрической оптимизации с применением методов визуализации // ИПМ им. М.В. Келдыша РАН, Россия, Москва, 2012.

25. Шрамм, Ф. К., Формозу, К.Т., Использование визуального интерактивного моделирования с использованием ПО по совершенствованию процесса принятия решений в системе обеспечения производства.

26. Попова, О. А., Технология извлечения и визуализации знаний на основе численного вероятностного анализа неопределенных данных // Информатизация и связь. 2013. № 2. С. 63–66.

27. Большаков, А. А., Керимов, Р. Н., Методы обработки многомерных данных и временных рядов // Москва, Горячая Линия — Телеком, 2007.

28. Бондарев, А. Е., Галактионов, В. А., Анализ и визуализация многомерных данных в задачах вычислительной газовой динамик // Programming and Computer Software. 2015. Т. 41. № 5. С. 247–252.

29. Булгаков, С. В., Агрегирование информационных моделей // Перспективы Науки и Образования, 2014, №3(9), С.9–13.

30. Васильев, В. Р., Волобой, А. Г., Вьюкова, Н. И., Галактионов, В. А.Контекстная визуализация пространственных данных // Препринты ИПМим. М. В. Келдыша. 2004. № 56. 23 с.

31. Горбань, А. Н., Методы нейроинформатики // КГТУ, Красноярск, 1998.

32. Добронец, Б. С., Попова, О. А., Гистограммный подходк представлению обработке данных космического и наземного мониторинга // Известия ЮФУ. Технические науки. 2014. № 6 (155). С. 14–22.

33. Добронец, Б. С., Попова, О. А., Численный вероятностный анализ неопределенных данных: монография // Сибирский федеральный университет, Красноярск, 2014 — 168 с.

34. Дугина, Т. О., Туннельная модель как способ представления совокупностей простых и многомерных данных // Известия высших учебных заведений. Поволжский регион. Технические науки, 2015. № 3 (35). — С. 5-14.

35. Зиновьев, А. Ю., Визуализация многомерных данных // КГТУ, Красноярск, 2000. 180 с.

36. Кудж, С. А., Цветков, В. Я., Информационные образовательные единицы // Дистанционное и виртуальное обучение, 2014. №1. С.24–31.

37. Масленников, О. П., Мильман, И. Е., Сафиуллин А. Э., Бондарев, А. Е., Низаметдинов, Ш. У., Пилюгин, В. В., Разработка и развитие системы интерактивного визуального анализа многомерных данных // Труды 25-й Международной Конференции по Компьютерной Графике и Зрению ГрафиКон'2015, Протвино, Россия, 22–25 сентября 2015 г., С. 227–231.

38. Методы создания стереоскопических изображений: Электронный ресурс URL: http://briefeducation.ru

39. Муха, В. С., Анализ многомерных данных: проблемы, состояние, перспективы // Доклады БГУИиР, п. Бровки, 6, Минск, р. Беларусь, 2003.

40. Погорелый, Е. С., Визуальное представление многомерных данных с использованием компьютерных моделей // Решетнёвские чтения: материалы XX Юбилейной междунар. науч.-практ. конф., посвящ. памяти генерального конструктора ракетно-космических систем академика М. Ф. Решетнева: в 2 ч. / под общ. ред. Ю. Ю. Логинова ; Сиб. гос. аэрокосмич. ун-т. — Красноярск, 2016. Ч. 2. С. 284–287.

41. Погорелый, Е. С., Представления многомерных данных для ВИМ-технологий // Журнал «Научные исследования и разработки молодых ученых». 2015. № 7. С. 120–124.

42. Погорелый, Е. С., Проблемный анализ визуализации многомерных данных // Международная научная конференция «Молодежь и наука: проспект Свободный». 2016.

43. Попова, О. А., Гистограммный информационно-аналитический подход к представлению и прогнозированию временных рядов // Информатизация и связь. 2014. № 2. С. 43–47.

44. Попова, О. А., Гистограммы второго порядка для численного моделирования в задачах с информационной неопределенностью // Известия ЮФУ. Технические науки, 06.2014.

45. Попова, О. А., Численный вероятностный анализ для агрегации, регрессионного моделирования и анализа данных // Информатизация и связь. 2015. № 1. С. 15–21.

46. Попова О. А., Численный вероятностный анализ для оптимизационных задач гидроэнергетики // Известия ИрГУ, 2015. Т. 12. Серия «Математика». С.79–92.

47. Попова, О. А., Першина, Е. Л., Чуканов, С. Н., Интеллектуальные системы поддержки принятия решений: комплексы программ, модели, методы, приложения // Федеральное агентство по образованию, ГОУ ВПО "Сибирская гос. автомобильно-дорожная акад. (СибАДИ)". Омск, 2011.

48. Тихонов, А. Н. Концепция сетецентрического управления сложной организационно-технической системой // Иванников, А. Д., Соловьёв, И. В., Цветков, В. Я., Кудж, С. А. — М.: МаксПресс, 2010. 136 с.

49. Цветков, В. Я. Информатизация: Создание современных информационных технологий. Часть 1. Структуры данных и технические средства // М., ГКНТ, ВНТЦентр, 1990. 118 с.

50. Dobronets, B. S., Krantsevich, A. M., Krantsevich, N. M. Software implementation of numerical operations on random variables // Журнал

Сибирского федерального университета. Серия: Математика и физика. 2013. Т. 6. № 2. С. 168–173.

51. Qingyu Zhang, Richard S. Segall, Mei Cao. Visual Analytics and Interactive Technologies: Data, Text and Web Mining Applications – New York: Information Science reference. 2011. – 363 c.

52. Dill J. Expanding the Frontiers of Visual Analytics and Visualization − London: Springer-Verlag 2012 – P. 555

53. Piatetsky-Shapiro, G., Machine Learning, Data Mining, and Knowledge Discovery: An Introduction // AAAI/MIT Press, 1996.

54. Scott, D. W., Multivariate density estimation: theory, practice and visualization // Rice University, Houston, Texas, 1993.

ПРИЛОЖЕНИЕ А

Полная версия программного модуля динамической визуализации 3D данных

```
{$reference System.Windows.Forms.dll}
{$reference System.Drawing.dll}


uses System, System.Drawing, System.Windows.Forms, OpenGL;

type Form1 = class (Form)

private
 _hdc : HDC;
 timer: Timer;
 components: System.ComponentModel.IContainer;

 R1   : double;
 R2   : double;
 R3   : double;
 R_m  : double;
 Z    : double;
 X    : double;
 Y    : double;
 Fi   : real;

public

//-----------------------------------------//
```

```
//------  Инициализация  приложения  -------//
//-----------------------------------------//
procedure Init;
begin
  // Настройка OpenGL
  glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LEQUAL);
  glClearDepth(1.0);
  glHint(GL_PERSPECTIVE_CORRECTION_HINT,GL_NICEST);

  // Настройка камеры/окна просмотра
  glMatrixMode( GL_PROJECTION );
  glLoadIdentity;
  glViewport(0, 0, Width, Height);
  gluPerspective(45.0, Width/Height, 0.1, 1000.0);

  // Установка матрицы мира/объекта
  glMatrixMode( GL_MODELVIEW );
  glLoadIdentity;

  // Инициализация переменных
  R1 :=  0.5;
  R2 :=  0.75;
  R3 :=  2;
  Z  := -2;
end;

//-----------------------------------------//
//------     Процедура рисование       -------//
//-----------------------------------------//
```

```
procedure Render(sender: Object; e: EventArgs);
var
  i    : integer;

begin
  // Заливка фона
  glClearColor(single(0.75), single(0.75), single(0.75), single(0.0));
  glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT);

  // Установка наблюдателя/камеры
  glLoadIdentity();
  gluLookAt(-7.0,0.0,7.0,  0.0,0.0,0.0,  0.0,1.0,0.0);

  glPushMatrix();

  // Рисуем объект
  glColor3f(0.125, 0.125, 0.125);
  glBegin(GL_LINES);
    glVertex3f(-2, -2, -2);
    glVertex3f(2, -2, -2);
  glEnd();
  glBegin(GL_LINES);
    glVertex3f(2, -2, -2);
    glVertex3f(2, 2, -2);
  glEnd();
  glBegin(GL_LINES);
    glVertex3f(2, -2, -2);
    glVertex3f(2, -2, 2);
  glEnd();
```

```
glColor3f(0.0, 0.7, 0.7);
if Z <= 2 then
begin
  glBegin(GL_LINE_LOOP); //квадрат
    glVertex3f(2, 2, Z);
    glVertex3f(2, -2, Z);
    glVertex3f(-2, -2, Z);
    glVertex3f(-2, 2, Z);
  glEnd;
end;

//КРАСНЫЙ-------------------------------------

glColor3f(1, 0.0, 0.0);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_1
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z);
  end;
glEnd;

glColor3f(0.95, 0.15, 0.15);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1.01));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_2
  for i := 0  to 27 do
  begin
```

```
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z-0.01);
  end;
glEnd;

glColor3f(0.9, 0.3, 0.3);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1.02));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_3
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z-0.02);
  end;
glEnd;

glColor3f(0.85, 0.45, 0.45);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1.03));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_4
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z-0.03);
  end;
glEnd;
```

```
glColor3f(0.8, 0.6, 0.6);
R_m := Sqrt(Sqr(R1) - Sqr(Z-1.04));
glBegin(GL_LINE_LOOP); //рисуем малую сферу_5
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) - 0.75;
    Y := R_m * cos(Fi) + 0.5;
    glVertex3f(X, Y, Z-0.04);
  end;
glEnd;

//ЖЕЛТЫЙ-------------------------------------

glColor3f(1, 1, 0.0);
R_m := Sqrt(Sqr(R2) - Sqr(Z+0.5));
glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_1
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z);
  end;
glEnd;

glColor3f(0.95, 0.95, 0.15);
R_m := Sqrt(Sqr(R2) - Sqr(Z+0.49));
glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_2
```

```
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z-0.01);
  end;
glEnd;

glColor3f(0.9, 0.9, 0.3);
R_m := Sqrt(Sqr(R2) - Sqr(Z+0.48));
glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_3
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z-0.02);
  end;
glEnd;

glColor3f(0.85, 0.85, 0.45);
R_m := Sqrt(Sqr(R2) - Sqr(Z+0.47));
glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_4
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z-0.03);
```

```
  end;
 glEnd;

 glColor3f(0.8, 0.8, 0.6);
 R_m := Sqrt(Sqr(R2) - Sqr(Z+0.46));
 glBegin(GL_LINE_LOOP); //рисуем среднюю сферу_5
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi) + 0.25;
    Y := R_m * cos(Fi) - 0.25;
    glVertex3f(X, Y, Z-0.04);
  end;
 glEnd;

//ЗЕЛЕНЫЙ-------------------------------------

 glColor3f(0.0, 1.0, 0.0);
 R_m := Sqrt(Sqr(R3) - Sqr(Z));
 glBegin(GL_LINE_LOOP); //рисуем большую сферу_1
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi);
    Y := R_m * cos(Fi);
    glVertex3f(X, Y, Z);
  end;
 glEnd;

 glColor3f(0.15, 0.95, 0.15);
```

```
R_m := Sqrt(Sqr(R3) - Sqr(Z-0.01));
glBegin(GL_LINE_LOOP); //рисуем большую сферу_2
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi);
    Y := R_m * cos(Fi);
    glVertex3f(X, Y, Z-0.01);
  end;
glEnd;

glColor3f(0.3, 0.9, 0.3);
R_m := Sqrt(Sqr(R3) - Sqr(Z-0.02));
glBegin(GL_LINE_LOOP); //рисуем большую сферу_3
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi);
    Y := R_m * cos(Fi);
    glVertex3f(X, Y, Z-0.02);
  end;
glEnd;

glColor3f(0.45, 0.85, 0.45);
R_m := Sqrt(Sqr(R3) - Sqr(Z-0.03));
glBegin(GL_LINE_LOOP); //рисуем большую сферу_4
  for i := 0  to 27 do
  begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi);
```

```
    Y := R_m * cos(Fi);
    glVertex3f(X, Y, Z-0.03);
   end;
  glEnd;

  glColor3f(0.6, 0.8, 0.6);
  R_m := Sqrt(Sqr(R3) - Sqr(Z-0.04));
  glBegin(GL_LINE_LOOP); //рисуем большую сферу_5
   for i := 0  to 27 do
   begin
    Fi := i * (Pi / 14);
    X := R_m * sin(Fi);
    Y := R_m * cos(Fi);
    glVertex3f(X, Y, Z-0.04);
   end;
  glEnd;

 //---------------------------------------------

  glPopMatrix();

  // Делаем шаг по оси Z
  Z += 0.01;
  if Z >= 2.05 then Z := -2;
  // Выводим содержимое на форму
  glFlush();
  SwapBuffers(_hdc);
end;

//---------------------------------------------//
```

```
//------     Процедура перерисовки     -------//
//-----------------------------------------//
procedure ResizeCl;
begin
  glMatrixMode( GL_PROJECTION );
  glLoadIdentity;
  glViewport(0, 0, Width, Height);
  gluPerspective(45.0, Width/Height, 0.1, 1000.0);
  glMatrixMode( GL_MODELVIEW );
  glLoadIdentity;
end;

//-----------------------------------------//
//------     Конструктор класса        -------//
//-----------------------------------------//
constructor Create;
begin
  // Берем указатель на окно
  _hdc := GetDC(self.Handle.ToInt32());
  // Создаем таймер
  components := new System.ComponentModel.Container;
  timer := new System.Windows.Forms.Timer(self.components);
  timer.Enabled := true;
  timer.Interval := 10;
  // Инициализируем OpenGL
  OpenGLInit(self.Handle);
  // Настройка OpenGL
  Init;
  // Вызов каждую 1миллисек проц. Render
  timer.Tick += Render;
```

```
        end;

        // Процедура рисования окна
        protected procedure OnPaint(e: System.Windows.Forms.PaintEventArgs);
override;
        begin
         //repeat
           Render(self, e);
         //until(false);
        end;

        // Процедура закрытия окна
        procedure Form_Closed(sender : object; e : EventArgs);
        begin
          // Уничтожения OpenGL
          OpenGLUninit(self.Handle);
        end;

        // Процедура выз. при изменения размера окна
        procedure Form_Resize(sender: object; e : EventArgs);
        begin
          ResizeCl;
        end;

        // Обработка нажатий кнопок
        procedure  Form_KeyDown(sender: object; e : KeyEventArgs  );
        begin

         //if (e.KeyCode = Keys.S ) then  RotY :=  RotY + 1;
         if (E.KeyCode = Keys.Escape) then self.Close;
```

```
end;

//-----------------------------------//
end; // end class


var f : Form1;
begin
  f := new Form1();
  f.Resize += f.Form_Resize;
  f.Closed += f.Form_Closed;
  f.KeyDown += KeyEventHandler(f.Form_KeyDown);

  f.Width  := 640;
  f.Height := 720;

  // FullScreen
  if (false) then begin
    f.TopMost := true;
    f.FormBorderStyle := System.Windows.Forms.FormBorderStyle.None;
    f.WindowState                                                    :=
System.Windows.Forms.FormWindowState.Maximized;
  end;

  // Запуск приложения
  Application.Run(f);

end.
```

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой СИИ
_____ Г. М. Цибульский
«___» _____ 2019 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Разработка модуля динамической визуализации 3D данных

09.04.02 Информационные системы и технологии

09.04.02.01 Информационно-управляющие системы

Руководитель	проф., д-р.физ.-мат. наук	Б. С. Добронец
Студент	КИ17-02-1/1М 031726495	Д. Б. Пархоменко
Рецензент	проф., д-р техн. наук	М.Н.Фаворская
Нормоконтролер	проф., д-р.физ.-мат. наук	Б. С. Добронец

Красноярск 2019

Заявление о согласии выпускника на размещение выпускной квалификационной работы в электронно-библиотечной среде ФГАОУ ВО СФУ

1 Я, Пархоменко Данил Борисович

фамилия, имя, отчество полностью

студент (ка) СФУ ИКИТ, КИ17-02-1М

институт/ группа

Федерального государственного автономного образовательного учреждения высшего образования «Сибирский федеральный университет» (далее – ФГАОУ ВО СФУ), разрешаю ФГАОУ ВО СФУ безвозмездно воспроизводить и размещать (доводить до всеобщего сведения) в полном объеме написанную мною в рамках выполнения образовательной программы

магистерской диссертации

указать выпускную квалификационную работу бакалавра, дипломную работу специалиста, дипломный проект специалиста, магистерскую диссертацию

на тему: Разработка модуля динамической визуализации 3D данных.

название работы

в открытом доступе в электронно-библиотечной среде (на веб-сайте СФУ), таким образом, чтобы любой пользователь данного портала мог получить доступ к выпускной квалификационной работе (далее – ВКР) из любого места и в любое время по собственному выбору, в течение всего срока действия исключительного права на выпускную работу.

2 Я подтверждаю, что выпускная работа написана мною лично, в соответствии с правилами академической этики и не нарушает авторских прав иных лиц.

«05» июля 2019

подпись

Рецензия

на магистерскую диссертацию студента СФУ Пархоменко Данила Борисовича на тему «Разработка модуля динамической визуализации 3D данных», представленную к защите по направлению 09.04.02 — Информационные системы и технологии по программе 09.04.02.01 — Информационно-управляющие системы

Актуальность обусловлена необходимостью динамической визуализации сложно-структурированных данных, данных научных экспериментов, многомерных данных.

Новизна работы заключается в использовании визуально-интерактивного моделирования для представления 3D данных.

Структура диссертационной работы. Работа состоит из введения, трех глав, заключения, список литературы содержит 31 использованный источник.

Результатом данной магистерской диссертации является разработка модуля динамической визуализации 3D данных с использованием стерео форматов.

Апробация. По результатам диссертации опубликованы две печатные работы.

Замечания и предложения по диссертации. В работе есть определенная доля опечаток и неточностей. Присутствует некоторая фрагментарность текста.

Заключение. Изложенный в работе теоретический, графический и демонстрационный материал оформлен в достаточной степени, в соответствии с требованиями нормативных документов СФУ. Работа выполнена в достаточно полном объеме, в соответствии с поставленной целью. Пархоменко Данил Борисович заслуживает присвоения квалификации «магистр» по направлению 09.04.02 — Информационные системы и технологии.

Оценка отлично.

Рецензент М.Н. Фаворская

профессор, доктор техн. наук,
Зав. кафедрой информатики и вычислительной техники СибГау

М. П. «27» июня 2019 г.

Подпись Фаворской М.Н. удостоверяю
Нач. отд. по раб
Начальник УК СибГАУ
27.06.2019
г.Красноярск