*Федеральное государственное бюджетное образовательное учреждение*

*высшего профессионального образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

**РАСЧЁТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту на тему:**

Разработка программы для работы с трёхмерными моделями молекул

Студент \_\_\_\_\_\_\_\_\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_**  Жарова Н. А.

(Подпись, дата) (Фамилия И. О.)

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_**Русакова З. Н.

(Подпись, дата) (Фамилия И. О.)

*2017 г.*

Оглавление

[Введение 4](#_Toc500905220)

[1 Аналитический раздел 7](#_Toc500905221)

[1.1 Анализ предметной области 7](#_Toc500905222)

[1.2 Обзор и анализ существующих программных систем и обоснование необходимости разработки. 8](#_Toc500905223)

[1.3 Обзор алгоритмов удаления невидимых линий и поверхностей 11](#_Toc500905224)

[1.3.1 Алгоритм Робертса 11](#_Toc500905225)

[1.3.2 Алгоритм, использующий z-буфер 11](#_Toc500905226)

[1.3.3 Алгоритм трассировки лучей 12](#_Toc500905227)

[1.4 Алгоритмы закрасок 12](#_Toc500905228)

[1.5 Выбор алгоритма удаления невидимых поверхностей и алгоритма закраски 13](#_Toc500905229)

[2 Конструкторский раздел 13](#_Toc500905230)

[2.1 Разработка алгоритма визуализации сфер 13](#_Toc500905231)

[2.2 Алгоритм визуализации цилиндров 16](#_Toc500905232)

[2.3 Алгоритм построения изображения 16](#_Toc500905233)

[2.4 Выбор структур данных 17](#_Toc500905234)

[3 Технологический раздел 17](#_Toc500905235)

[3.1 Выбор технологии программирования 18](#_Toc500905236)

[3.2 Выбор среды и языка программирования 18](#_Toc500905237)

[3.3 Диаграммы разработанных классов 18](#_Toc500905238)

[3.3.1 Диаграмма классов визуализации 18](#_Toc500905239)

[3.3.2 Диаграмма классов для хранения структуры молекулы 19](#_Toc500905240)

[3.4 Интерфейс пользователя и порядок работы с программой 20](#_Toc500905241)

[3.5 Входные и выходные данные 22](#_Toc500905242)

[4 Экспериментально – исследовательский раздел 22](#_Toc500905243)

[4.1 Основная программа визуализации молекулярных структур. 22](#_Toc500905244)

[4.2 Программа визуализации молекулярных структур с использованием OpenGL 23](#_Toc500905245)

[4.3 Программа визуализации молекулярных структур с использованием OpenGL и шейдера сфер. 24](#_Toc500905246)

[Заключение 26](#_Toc500905247)

[Список использованных источников 27](#_Toc500905248)

# Введение

Исследования в области химии и биологи неизменно связаны с изучением микроскопических объектов (атомов и молекул), которые нельзя увидеть без использования специального оборудования. В частности, на данный момент проводится большое количество экспериментов направленных на определение различных белковых структур, из которых состоят все живые организмы. Данные, полученные в результате экспериментов, в основном представляют собой огромные текстовые файлы, в которых подробно описывается положение и тип каждого атома, а также некоторые другие характеристики, которые позволят точно описать структуру изучаемого соединения. Человеку очень сложно, взглянув на такой файл, сразу понять, что в нём описывается, поэтому ему на помощь приходят специальные программы, которые позволяют визуализировать содержимое этих файлов. Кроме того, иллюстрации, созданные при помощи таких программ, широко используются при написании различных научных работ, печатаются в учебниках, научных журналах и методических пособиях. [1-3]

Целью данной работы является разработка эффективного алгоритма визуализации молекулярных и кристаллических структур, а также программы, использующей этот алгоритм и удовлетворяющей следующим требованиям:

1. Интерфейс программы должен быть простым и понятным
2. Программа должна обрабатывать файлы формата PDB и генерировать изображение, соответствующее содержанию файла
3. Программа должна предоставлять возможность экспорта полученного изображения в форматы JPEG и PNG
4. Программа должна предоставлять возможность поворачивать и масштабировать модель белкового соединения, считанную из файла

Для достижения поставленной цели необходимо:

1. провести исследование существующих алгоритмов визуализации молекулярных и кристаллических структур;
2. на основе полученных данных, разработать эффективный алгоритм
3. провести тестирование, разработанного алгоритма;
4. разработать дружественный интерфейс для работы с визуализируемой моделью.

Программное обеспечение (ПО) для визуализации результатов квантово-химических экспериментов позволяет в более удобной для человека форме представлять структуру молекул. Изучение этой структуры, позволяет научным работникам делать предположения о свойствах молекул и, как результат, определять дальнейшее направление исследований.

Кроме того, ПО для визуализация молекулярных структур используется для публикации научных работ и статей, связанных с химией и молекулярной биологией. Использование полученных с помощью рассматриваемого ПО изображений значительно увеличивает читаемость текста и позволяет без дополнительных усилий представить молекулу или кристалл, о котором идёт речь.

Не стоит забывать и о такой важной сфере, как образование. В каждом учебнике по химии приводятся изображения тех или иных молекулярных структур, полученных с помощью программ визуализации молекул. Такие иллюстрации помогают ученикам представить молекулу и дают учителям возможность на примере показать, чем занимается химия.

Визуализация результатов эксперимента является важной частью любой научной работы. Таким образом, целью данного курсового проекта является написание программы для визуализации и редактирования кристаллических и молекулярных структур в виде шаростержневых моделей, которая в качестве входных данных получает файл формата .pdb и позволяет пользователю сохранить результат визуализации в виде изображения формата .png или .jpeg.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ предметной области.
2. Провести анализ существующих методов, позволяющих достичь поставленной цели, и выбрать наиболее оптимальные методы.
3. Разработать и реализовать программу для визуализации молекулярных и кристаллических структур.
4. Проверить работоспособность программной реализации.

# 1 Аналитический раздел

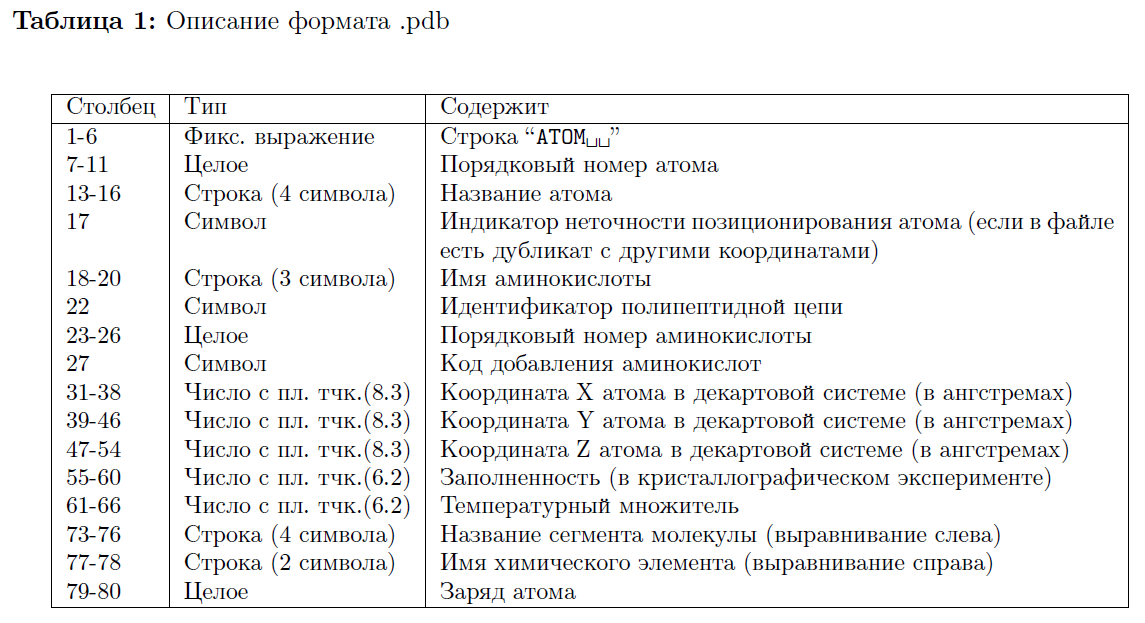
Данный раздел посвящен анализу предметной области и задачам построения изображения молекулярных и кристаллических структур. Здесь также проводится обзор существующих методов и алгоритмов, позволяющих решить эти задачи.

* 1. Анализ предметной области

Не смотря на активное развитие хемоинформатики, в том числе в сфере молекулярного моделирования, русскоязычной литературы, в которой бы излагались и систематизировались методы и подходы к визуализации трехмерных моделей молекулярных структур, не так много. Большинство источников, связанных с данной тематикой, представляют собой работы иностранных авторов, которые не издаются на русском языке, русскоязычные книги по молекулярной динамике отсутствуют в свободном доступе, а статьи в основном имеют узкую специализацию, что не позволяет использовать их в качестве подспорья для изучения общих алгоритмов визуализации различных молекулярных структур. Таким образом, одной из главных задач в данной работе становится поиск и исследование существующих подходов к построению трехмерной модели молекул и последующей визуализации.

Начать следует с того, что основными входными данными для такого рода задач являются файлы формата PDB из международных баз данных белковых структур (например, <http://www.wwpdb.org/>). Описание формата .pdb представлено в таблице 1 [4]. Следует заметить, что часто файлы не полностью соответствуют данному формату.

Отсюда следует, что в качестве основных объектов при построении трехмерной модели будут атомы с характеристиками, содержащимися в соответствующей строке файла PDB. Молекулярная структура будет состоять из массива атомов и связей между ними. При визуализации в виде шаростержневой модели, каждому атому будет соответствовать сфера с центром в точке, заданной координатами атома, радиус и цвет сферы будут зависеть от типа атома. А каждой связи будет соответствовать цилиндр, соединяющий сферы, связанных атомов.



1.2 Обзор и анализ существующих программных систем и обоснование необходимости разработки.

Число программ, решающих задачу визуализации молекулярных структур с каждым годом становится всё больше. В подавляющем большинстве случаев это относительно небольшие программы, созданные под конкретные нужды небольшой группы учёных, качество получаемых в этих программах изображений остаётся достаточно низким, а сами проекты не получают дальнейшего развития. Однако хотелось бы обратить внимание на QuteMol[5] – это программа с открытым исходным кодом, несмотря на то, что последняя версия появилась в свободном доступе в 2007 году, в ней используется широкий набор визуальных эффектов, позволяющих получить высококачественное изображение молекул. В качестве входных данных используются файлы формата PDB, экспортировать полученные изображения можно в форматы PNG, JPEG и GIF. В программе представлено 15 режимов представления изображения (рис. 1) и 3 типа представления молекул: Space-Fill, Licorice, Balls'n'Sticks. Простой и понятный интерфейс, а также ряд настроек позволяют с лёгкостью получить изображение, отвечающее текущим требованиям. Однако, у данной программы есть ряд проблем, связанных с построением модели молекулярной структуры из файлов PDB, из-за чего некоторые белковые структуры отображаются некорректно.

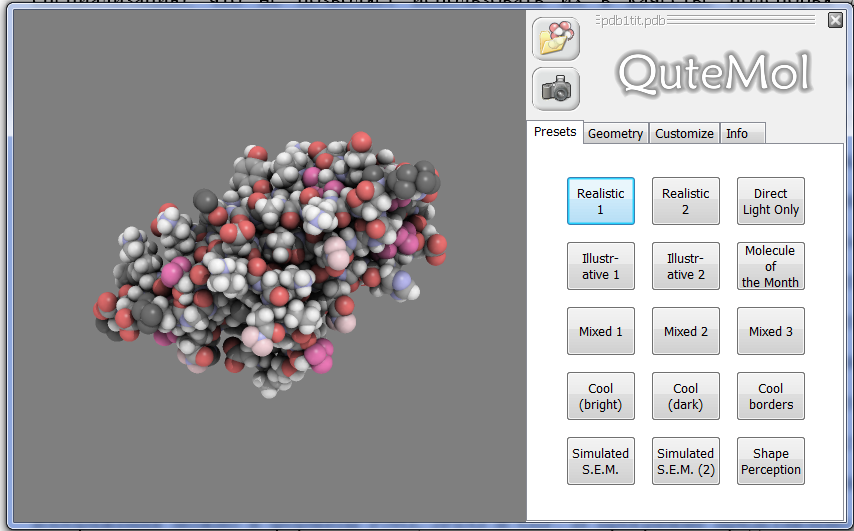


Рисунок 1. Режимы представления изображения в QuteMol. Изображена структура одного из доменов титина с индексом в базе данных белковых структур 1TIT

Совершенно на другом уровне находится проект VMD [6]. Первая версия программы появилась в 1995 году, но активная работа над проектом ведётся и в настоящее время. Это один из лучших на данный момент инструментов для визуализации, анализа и анимации больших биомолекулярных систем. На сайте проекта доступен исходный код программы, кроме того имеется большое количество документации как для пользователей, так и разработчиков. Несмотря на то, что программа имеет ряд преимуществ перед своими аналогами: широкие возможности анализа биомолекулярных систем, синтезирования структур, построения файлов PSF, в которых хранится структура белка, информация о типах атомов и их связности - интерфейс программы довольно неудобен и не является интуитивно понятным, а качество синтезируемого изображения значительно уступает QuteMol.

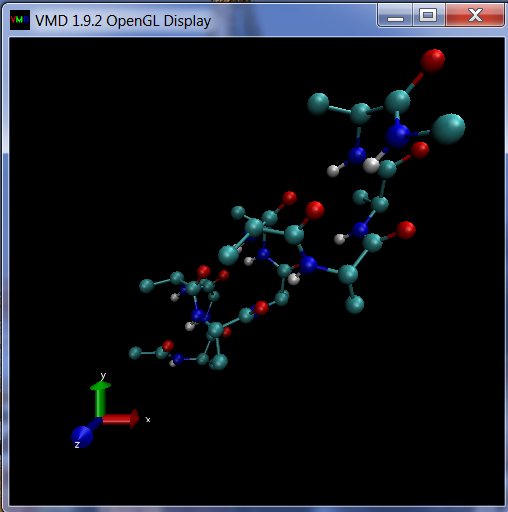


Рисунок 2. Изображение молекулярной структуры аланина сгенерированное в программе VMD.

Несмотря на многообразие существующих решений, рабзработка программ визуализации молекулярных и кристаллических структур остаётся актуальной, так как с каждым годом улучшаются характеристики графических процессоров, появляются новые алгоритмы компьютерный графики, позволяющие сократить время, затрачиваемое на построение изображения, и повышающее качество изображения. Кроме того, хотелось бы разработать программу, позволяющую не только быстро и качественно визуализировать загруженную из файла молекулярную структуру, но и дающую возможность пользователю, не обладающему специальными знаниями (например, о формате файла PDB), «на лету» редактировать эту структуру.

1.3 Обзор алгоритмов удаления невидимых линий и поверхностей

Одним из важнейший алгоритмов в трёхмерной графике является алгоритм удаления невидимых линий и поверхностей, так как он в значительной мере влияет на производительность программы при отрисовке большого числа объектов. Ниже рассмотрены наиболее распространенные алгоритмы.

1.3.1 Алгоритм Робертса

В соответствии с этим алгоритмом прежде всего удаляются из каждого тела те ребра или грани, которые перекрываются самим телом. Затем каждое из видимых ребер каждого тела сравнивается с каждым из оставшихся тел для определения того, какая его часть или части, если таковые есть, перекрываются этими телами [7].

Преимущества данного алгоритма в том, что математические методы, используемые в нем, просты, мощны и точны. Недостаток заключается в том, что вычислительная трудоемкость алгоритма Робертса растет теоретически, как квадрат числа объектов. Реализация оптимизированных алгоритмов весьма сложна. Кроме того, этот алгоритм может использоваться только с выпуклыми объектами.

1.3.2 Алгоритм, использующий z-буфер

Данный алгоритм удаления невидимых поверхностей является одним из простейших. Этот алгоритм работает в пространстве изображения. Здесь наряду с буфером кадра вводится Z-буфер, представляющий собой буфер глубины, в котором запоминаются координаты Z (глубина) каждого видимого пикселя в пространстве изображения. В процессе работы глубина каждого нового пикселя, который надо занести в буфер кадра, сравнивается с глубиной того пикселя, который уже занесен в Z-буфер. Если это сравнение показывает, что новый пиксель расположен ближе к наблюдателю, чем пиксел, уже находящийся в буфере кадра, то новый пиксель заносится в буфер кадра, помимо этого производится корректировка Z-буфера: в него заносится глубина нового пикселя. Если же глубина нового пикселя меньше, чем глубина хранящаяся в буфере, то никаких действий не производится.

1.3.3 Алгоритм трассировки лучей

В этом методе траектория каждого луча, исходящего от наблюдателя, отслеживается, чтобы определить, какие именно объекты сцены пересекаются с данным лучом. Необходимо проверить пересечение каждого объекта сцены с каждым лучом. Если луч пересекает объект, то определяются все возможные точки пересечения луча и объекта. Эти пересечения упорядочиваются по глубине. Пересечение с максимальным значением Z представляет видимую поверхность для данного пиксела.

К достоинствам данного алгоритма можно отнести возможность изображения гладких объектов без аппроксимации их примитивами (например, треугольниками) и реалистичность получаемых картин. Серьёзным недостатком алгоритма трассирования является низкая производительность.

1.4 Алгоритмы закрасок

Если при построении полигональной поверхности для каждой грани используется по одной нормали, то модель освещения создает изображение, состоящее из отдельных многоугольников. Для получения сглаженного изображения необходимо использовать один из алгоритмов закраски.

При закраске Гуро сначала определяется интенсивность вершин многоугольника, а затем с помощью билинейной интерполяции вычисляется интенсивность каждого пиксела на сканирующей строке. Этот метод хорошо сочетается с простой моделью освещения, в которой учитывается диффузная составляющая.

При закраске Фонга вдоль сканирующей строки интерполируется вектор нормали. Затем он используется в модели освещения для вычисления интенсивности пиксела.

1.5 Выбор алгоритма удаления невидимых поверхностей и алгоритма закраски

Для удаления невидимых поверхностей выбран алгоритм, использующий Z-буфер, поскольку он прост в реализации и позволяет удалять сложные поверхности и визуализировать пересечения таких поверхностей. Алгоритм трассировки лучей не подходит, так как предполагается визуализация большого количества объектов (несколько тысяч сфер и цилиндров).

Закраска будет осуществляться по методу Фонга, поскольку он дает лучшую локальную аппроксимацию кривизны поверхности и, следовательно, получается более реалистичное изображение.

# 2 Конструкторский раздел

В данном разделе подробно рассмотрены алгоритмы визуализации сфер и цилиндров, дано описание используемых типов и структур данных.

2.1 Разработка алгоритма визуализации сфер

Так как данная программа предназначена для качественной визуализации больших молекулярных систем (т.е. необходимо с большой скоростью визуализировать тысячи сфер), было принято решение отказаться от классического построения сфер, основанного на вычислении дополнительных надстроек на гранях правильных полиэдров [8]. Несмотря на то, что методы построения сфер, предложенные в [8], являются эффективными, они аппроксимируют сферу треугольниками, то есть при дальнейшей отрисовке сфер, необходимо выполнить закраску каждого треугольника, а это долго. В качестве альтернативы был разработан следующий алгоритм визуализации сфер в пространстве изображения:

1. Вычисление экранных координат центра сферы
2. Вычисление радиуса сферы в пространстве изображения
3. Применение модифицированного алгоритма генерации окружности Брезенхема.

За основу модифицированного алгоритма был взят алгоритм, приведённый в [9]. В данном алгоритме так же учитывается то, что направление нормали поверхности сферы в данной точке можно вычислить как вектор, направленный из центра сферы к данной точке на поверхности сферы. Модификация алгоритма генерации окружности Брезенхема для рисования сфер:

1. Ввод исходных данных R (радиус окружности) и Xc,Yc (координаты центра окружности), Color (цвет сферы)
2. Задание начальных значений текущих координат пикселя X=0, Y=R, параметра D=2(1-R), установка конечного значения ординаты пикселя Yk=0.
3. Для всех пикселей в строке [(Х,0);(X,Y)]:
   1. Вычислить глубину Z = sqrt(X\*X + Y\*Y)
   2. Вычислить нормированный вектор нормали N = normalize(X,Y,Z)
   3. Закрасить пиксель (X+Xc,Y+Yc) цветом color(Color, N)
   4. N.X = -N.X
   5. Закрасить пиксель (-X+Xc,Y+Yc) цветом color(Color, N)
   6. N.Y = -N.Y
   7. Закрасить пиксель (-X+Xc,-Y+Yc) цветом color(Color, N)
   8. N.X = -N.X
   9. Закрасить пиксель (X+Xc,-Y+Yc) цветом color(Color, N)
4. Проверка окончания работы: если Y<Yk, то переход к п.11.
5. Анализ значения параметра D: если D<0, то переход к п.6; если D=0, то переход к п.9; если D>0, то переход к п.7.
6. Вычисление параметра D1=2D+2Y-1 и анализ полученного значения: если D1<0, то переход к п.8; если D1>0, то переход к п.9.
7. Вычисление параметра D2=2D-2X-1 и анализ полученного значения: если D2<0, то переход к п.9; если D2>0, то переход к п.10.
8. Вычисление новых значений X и D (горизонтальный шаг): X=X+1; D=D+2X+1. Переход к п.3.
9. Вычисление новых значений X,Y и D (диагональный шаг): X=X+1; Y=Y-1; D=D+2(X-Y+1). Переход к п.3.
10. Вычисление новых значений Y и (вертикальный шаг): Y=Y-1; D=D-2Y+1. Переход к п.3.
11. Конец.

N.X, N.Y, N.Z – соответственно x, y, z координаты вектора нормали N.

Функция normaize(N) возвращает вектор N единичной длины.

Функция color(Color, N) возвращает цвет пикселя, вычисленный в соответствии с моделью освещения по Фонгу. Интенсивность пикселя будет рассчитываться с использованием простой модели освещенности (1).

(1)

где - интенсивность отраженного света, - интенсивность рассеянного света, - коэффициент диффузного отражения рассеянного света, - интенсивность точечного источника, - коэффициент диффузного отражения, — угол между направлением света и нормалью к поверхности, – коэффициент зеркального отражения, — степень, аппроксимирующая пространственное распределение зеркально отраженного света, α – угол между направлением взгляда наблюдателя и вектором отражения.

Преимуществом данного алгоритма визуализации сфер является скорость. Нормаль, единичной длины к поверхности сферы, по сути вычисляется один раз для четырех точек, для корректного расчёта интенсивности в конкретной точке меняются только знаки компонент X и Y. Кроме того, так как данный алгоритм работает с экранными координатами, мы всегда выбираем положительное значение Z, таким образом отсекая невидимую половину сферы.

В качестве недостатка можно выделить потерю точности при нахождении глубины

2.2 Алгоритм визуализации цилиндров

Этот алгоритм аналогичен предыдущему:

1. Вычисление экранных координат центров сфер, соединяемых цилиндром.
2. Применение модифицированного целочисленного алгоритма построения отрезка Брезенхема.

Примечание: радиус цилиндра в экранной системе координат вычисляется один раз, перед отрисовкой первого цилиндра.

За основу модифицированного алгоритма был взят алгоритм, приведённый в [9]. В данном алгоритме учитывается то, что направление нормали поверхности цилиндра в данной точке можно вычислить как вектор, направленный от оси цилиндра к данной точке на поверхности цилиндра и перпендикулярный оси.

Разработанная модификация представляет собой по сути два вызов алгоритма Брезенхема построения отрезка. Сначала он выполняется для отрезка, перпендикулярного оси цилиндра, длиной равной удвоенному радиусу цилиндра, причём построение отрезка выполняется сразу в двух противоположных направлениях от одного из концов оси цилиндра. Найденные точки отрисовываются и запоминаются в специальный массив. В отдельный массив запоминаются цвета пикселей. После этого второй раз выполняется алгоритм Брезенхема уже вдоль оси цилиндра, причём смещаются и отрисовываются все точки, ранее запомненные в массив, соответствующим цветом из массива цветов.

2.3 Алгоритм построения изображения

Формальное описание алгоритма с использованием Z-буфера будет выглядеть следующим образом:

1. Заполнить буфер кадра фоновым значением цвета.
2. Заполнить значение Z-буфер минимальным значением глубины Z.
3. Для каждого визуализируемого пикселя (X,Y):
   1. Если глубина Z(X,Y) текущего пикселя больше значения Z-буфер[X][Y], то Z-буфер[X][Y] = Z(X,Y); в буфере кадра заменить значение в позиции (X,Y) на цвет визуализируемого пикселя.

2.4 Выбор структур данных

Для хранения Z-буфера выбран массив типа float, такой точности достаточно для хранения глубины.

Для хранения буфера кадра выбран массив типа unsign integer.

Для хранения трёхмерных координат используется структура, содержащая 3 поля типа float.

Сфера описывается структурой, содержащей структуру для хранения трёхмерных координат центра сферы, поле типа float для радиуса и поле типа unsign integer для цвета.

Цилиндр описывается структурой, которая содержит две структуры для хранения трехмерных координат концов оси цилиндра, поле типа float для радиуса и поле типа unsign integer для цвета.

Для визуализации будут использоваться следующие структуры данных:

* сцена: список с произвольным числом графических объектов
* графический объект: сфера или цилиндр
* камера: положение в трехмерном пространстве
* источник света: положение в трехмерном пространстве, интенсивность

# 3 Технологический раздел

Этот раздел содержит обоснованный выбор средств программной реализации, информацию о входных и выходных данных, диаграмму разработанных классов, описание интерфейса пользователя и порядок работы с программой.

3.1 Выбор технологии программирования

Программа будет реализована с использованием объектно-ориентированного подхода, так как классы позволяют разрабатывать программы на основе информационной модели, абстрагируясь от реализации, инкапсуляция защищает данные от случайных изменений, наследование позволяет довольно легко добавлять функциональность, не изменяя ранее разработанные методы, а полиморфизм предоставляет возможность использовать объекты разных классов через один интерфейс. Всё это удобно использовать в текущем проекте.

3.2 Выбор среды и языка программирования

Для реализации программы используется выбран язык С++, поскольку это универсальный язык, обладающий высокой вычислительной производительностью. Кроме того, он поддерживает парадигму ООП.

В качестве среды разработки выбрана среда Qt, поскольку она предоставляет удобные инструменты для разработки пользовательского интерфейса, а также некоторые классы-контейнеры, которые удобно использовать.

3.3 Диаграммы разработанных классов

3.3.1 Диаграмма классов визуализации

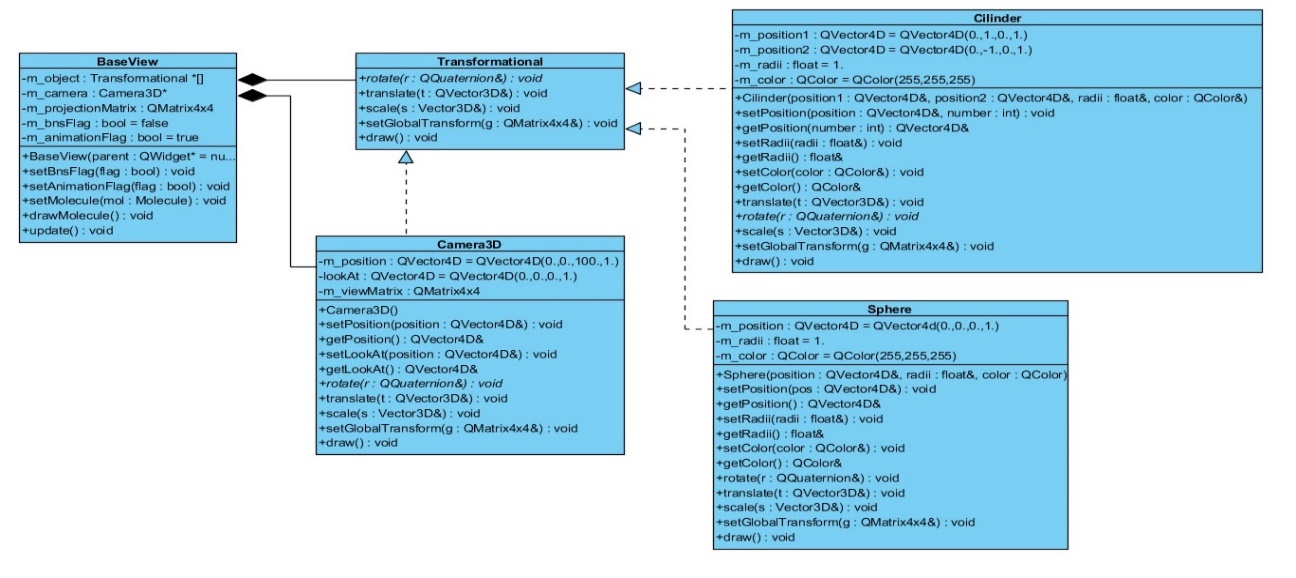


Рисунок 3. Диаграмма классов визуализации

BaseView – основной класс визуализации содержит список указателей на графические объекты (сферы и цилиндры), которые будут визуализированы при вызове метода draw(). Кроме того содержит указатель на камеру.

Transformational – базовый абстрактный класс для графических объектов, реализует интерфейс.

Sphere – класс для хранения информации о сферах.

Cilinder – класс для хранения информации о цилиндрах.

Camera3D - класс для хранения и формирования матрицы преобразования к системе координат наблюдателя.

3.3.2 Диаграмма классов для хранения структуры молекулы

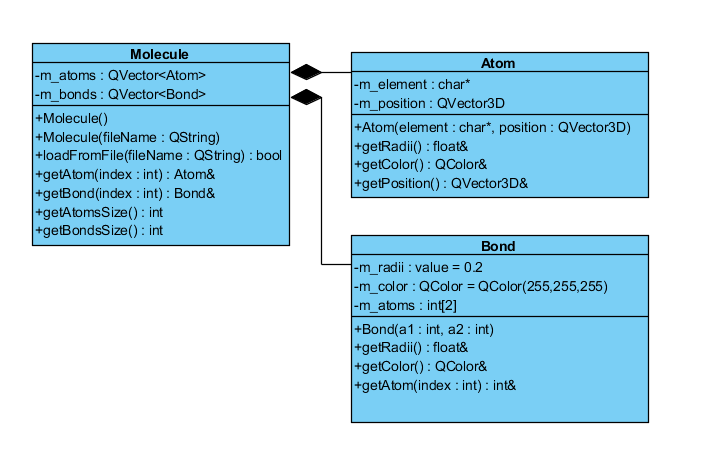


Рисунок 4. Диаграмма классов для хранения структуры молекулы

Molecule – класс для хранения молекулярных и кристаллических структур. Для загрузки модели из файла используется метод loadFromFile.

Atom – класс для хранения информации об атоме.

Bond – класс для хранения информации о ковалентной связи. В массиве m\_atoms хранятся индексы атомов, соединённых данной ковалентной связью, в массиве атомов объека класса Molecule.

3.4 Интерфейс пользователя и порядок работы с программой

Главное окно пользовательского интерфейса, изображенное на рис.5, содержит панель меню, рабочее пространство для визуализации и редактирования молекул. Через панель меню осуществляется загрузка молекулярной структуры из файла, сохранение результатов работы программы, добавление, редактирование и удаление атомов, добавление, редактирование и удаление ковалентных связей, выбрать представление молекулы (шаростержнвое или заполнящее пространство), включить/выключить анимацию.

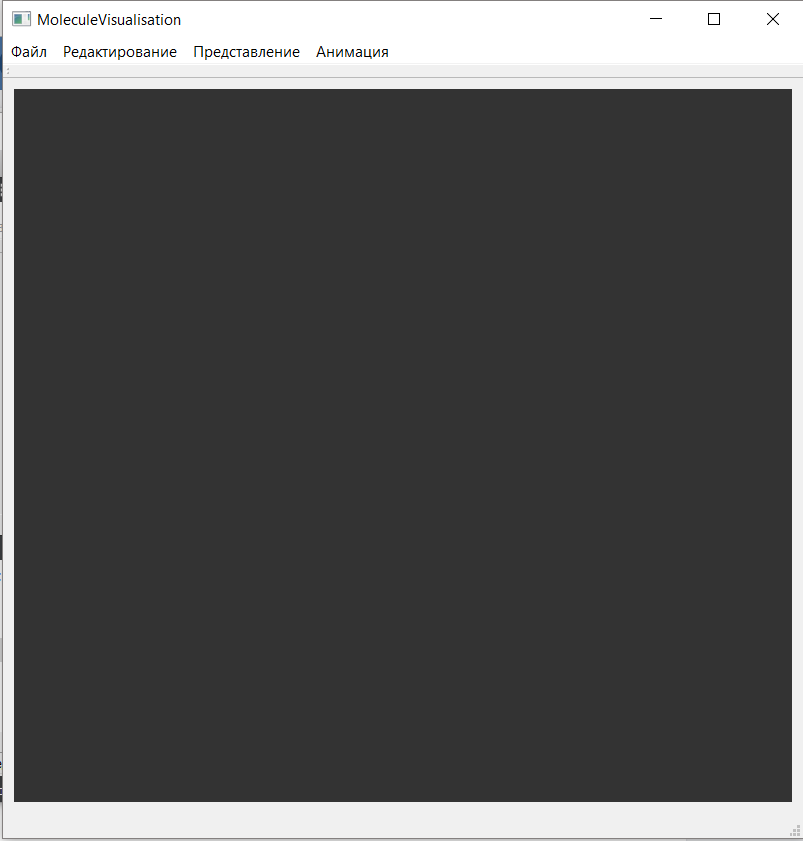


Рисунок 5. Главное окно

Для начала работы необходимо выбрать файл с молекулярной или кристаллической структурой и загрузить его в программу. Это можно сделать через панель меню (Файл −> Открыть) или с помощью горячей клавиши CTRL+O. Выбранная молекула появится в рабочей области пользователя.

Начать редактирование атомов или ковалентных связей можно, выбрав на панели меню Редактирование ­−> Редактировать. Появится диалоговое окно, изображённое на рисунке 6. Для редактирования атомов надо выбрать вкладку «Атом». С помощью поля «Номер атома» можно переключаться между атомами. А остальные поля позволяют менять все характеристики атома. Изменения вступят в силу после нажатия кнопки «Ок». Для редактирования ковалентных связей надо выбрать вкладку «Ковалентная связь». С помощью поля «Номер ковалентной связи» можно переключаться между ковалентными связями.

Начать создание атома/связи можно, выбрав на панели меню Редактирование −> Добавить. Появится диалоговое окно, изображённое на рисунке 6, с заблокированными полями «Номер атома» и «Номер ковалентной связи». После нажатия на кнопку «Ок» изменения вступят в силу. Для удаления атома в окне «Редактирование» на вкладке «Атом» необходимо нажать кнопку «Удалить». Для удаления связи в окне «Редактирование» на вкладке «Ковалентная связь» необходимо нажать кнопку «Удалить».

Результат работы программы можно сохранить, выбрав на панели меню действие Файл –> Сохранить или с помощью горячей клавиши CTRL+S. Изображение молекулы будет сохранено в указанной папке в формате png или jpeg.

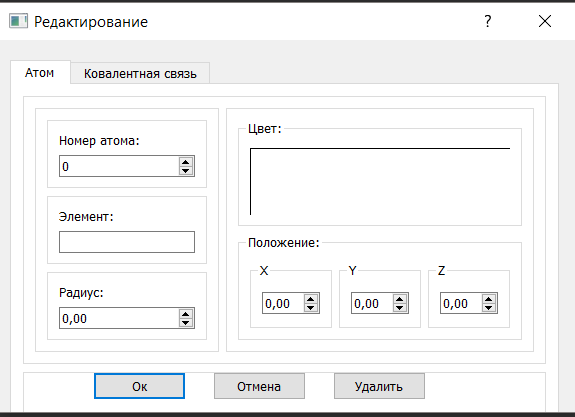
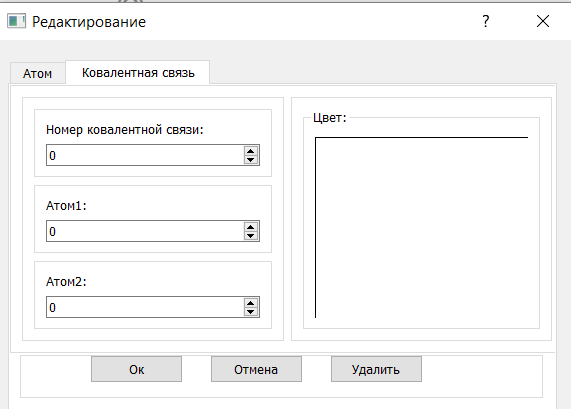
 

Рисунок 6. Редактирование атома. Редактирование ковалентной связи.

3.5 Входные и выходные данные

В качестве входных данных в программу загружается молекулярная или кристаллическая структура в файле формата PDB.

Выходными данными являются изображения в формате PNG или JPEG.

# 4 Экспериментально – исследовательский раздел

В данном разделе приводятся результаты исследования, цель которого – оценить скорость и качество визуализации программы. По возможности найти способы улучшения этих характеристик.

* 1. Основная программа визуализации молекулярных структур.

На рисунке 7 приведены результаты работы основной программы. Качество изображения приемлемое, однако при визуализации молекул с количеством атомов, превышающим 1000, становятся заметны небольшие задержки при движении молекулы.

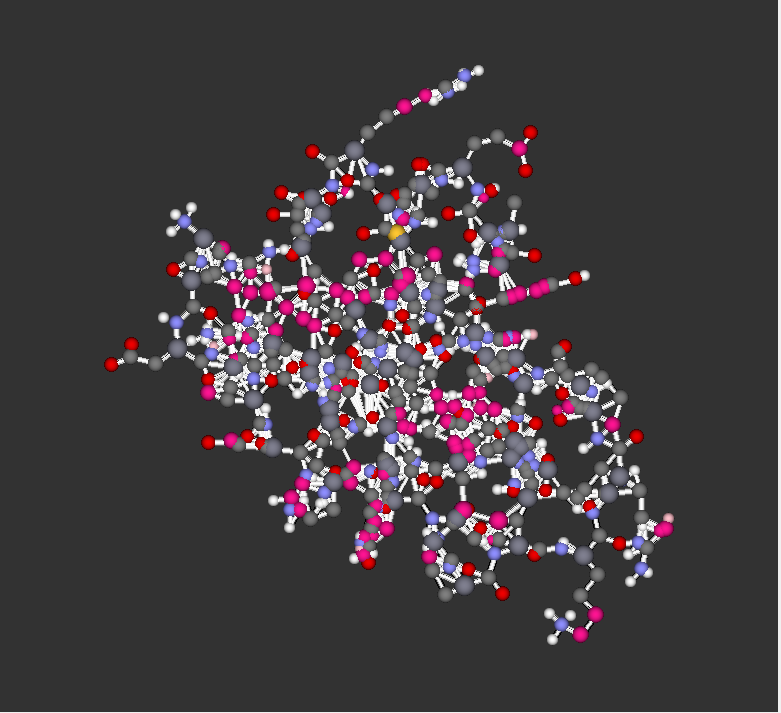
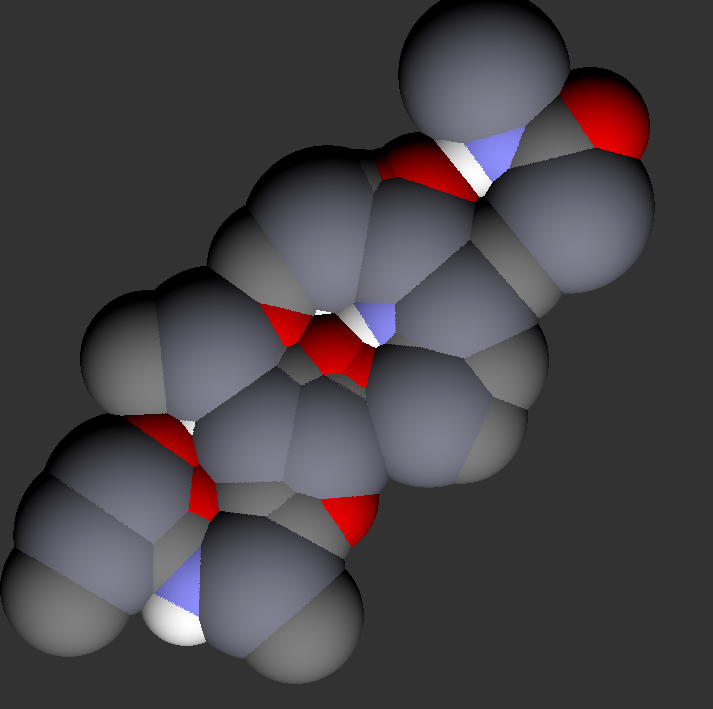
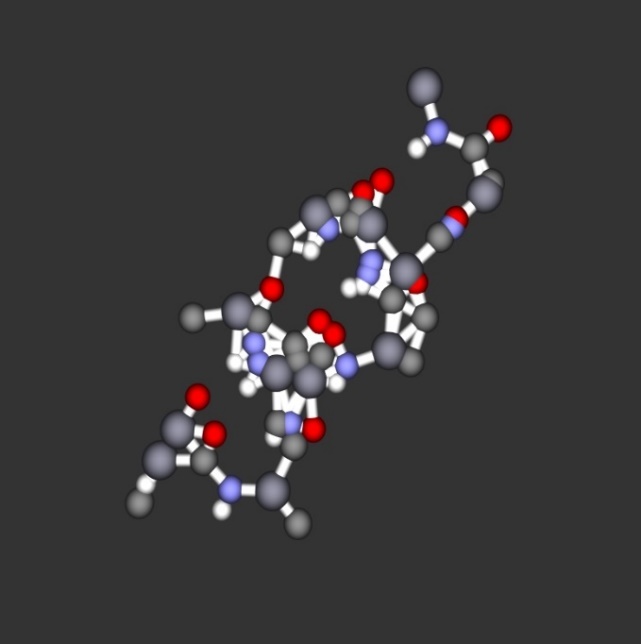
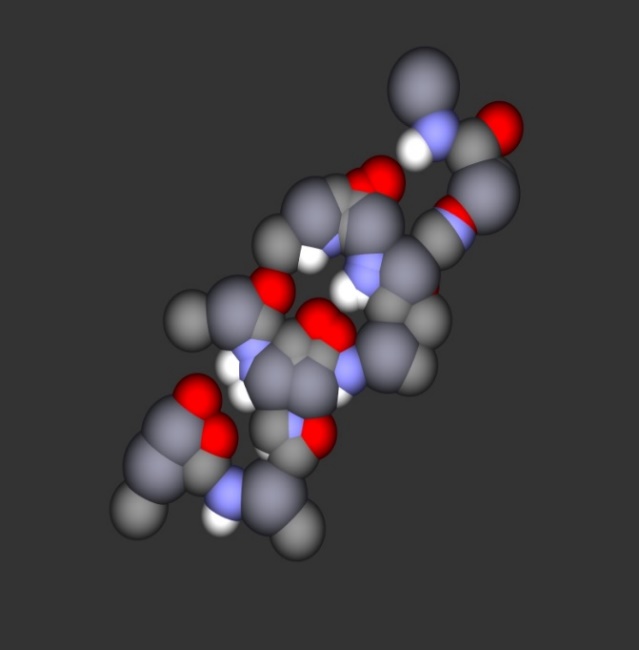


Рисунок 7. Результаты визуализации основной программы

Зависимость времени построения кадра от количества атомов приведена в гистограмме 1. Полученные результаты объясняются тем, что алгоритмы работают в пространстве изображения. Таким образом, время визуализации зависит не только от количества атомов в молекуле, но и от её структуры. Если большое число атомов и связей экранируются другими атомами и связями, то время визуализации значительно уменьшается. Кроме того, время визуализации зависит от текущего масштаба. Чем меньше масштаб, тем меньше время визуализации.

4.2 Программа визуализации молекулярных структур с использованием OpenGL

Перенос процесса визуализации на графический процессор может значительно ускорить работу приложения [3-4]. Поэтому на основе разработанной программы, была реализована программа с поддержкой OpenGL. Qt предоставляет удобные классы для работы с этим API. Кроме того, были написаны вершинный и фрагментный шейдеры на языке GLSL ES 3.2. Этот стандарт обратно совместим почти со всеми старыми версиями и позволяет писать шейдеры, которые будут правильно выполнятся на большинстве графических процессоров[11]. Данная программа была написана с целью ознакомления с современными средствами работы с трёхмерной графикой, а также получение сравнительных характеристик основной программы. В данном случае, сферы и цилиндры будут аппроксимироваться большим количеством полигонов, будет использовать стандартный Z-буффер OpenGL, модель освещения по Фонгу реализована в фрагментном шейдере.

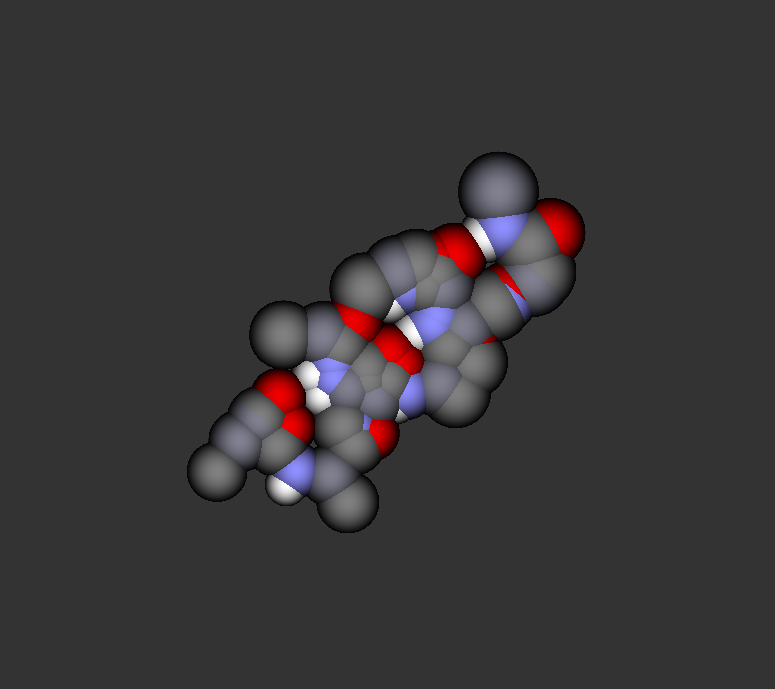


**Рисунок 8. Результаты визуализаци молекулярных структур с использованием OpenGL.**

Качество изображения улучшилось, при движении визуализации более 1000 атомов заметна задержка при движении молекулы. Временные характеристики приведены в гистограмме 2. Из гистограммы видно, что эффективность данной реализации выше предыдущей при малом количестве молекул (до 1000), и ниже при большом количестве молекул

4.3 Программа визуализации молекулярных структур с использованием OpenGL и шейдера сфер.

Аппроксимировать сферы и цилиндры большим количеством треугольников неэффективно. Поэтому возник вопрос, как можно отрисовывать сферы с использованием шейдеров, сохранив высокое качество изображения и увеличив производительность. В ходе изучения документации OpenGL [11] выяснилось, что можно менять размер точек и что в фрагментном шейдере можно не отрисовывать текущий пиксель вызовом discard. В результате был разработан новый фрагментный шейдер: в нём проверяется, принадлежит ли текущий пиксель окружности, и если принадлежит, то выполняется расчёт цвета пикселя, в противном случае пиксель не отрисовывется.



**Рисунок 8. Результаты визуализаци молекулярных структур с использованием OpenGL и нового шейдера.**

Как видно из рисунка 8, качество изображения почти не изменилось, однако прирост производительности составил при 4000 атомов порядка 560%

# Заключение

Разработана программа визуализации молекулярных и кристаллических структур в соответствии с выданным техническим заданием. Для этого проведен анализ существующих программных систем и методов визуализации, на их основе разработаны алгоритмы построения сфер и цилиндров. Эти алгоритмы работают в пространстве изображения и имеют высокую производительность. Разработан пользовательский интерфейс, позволяющий загружать в программу и редактировать молекулярные структуры, сохранять результат работы программы в виде изображений. Проведено исследование методов визуализации трехмерных сцен с использованием OpenGl. В результате проведённых экспериментов, получены временные характеристики разработанных программ и их сравнение.

Данная программа может использоваться в качестве средства визуализации и простого редактирования молекулярных и кристаллических структур.

# Список использованных источников

1. Порозов, Ю. Б. Биоинформатика и средства компьютерного анализа и визуализации макромолекул. / Ю. Б. Порозов // Саратовский научно-медицинский журнал – 2010 - Т. 6, № 2 - С. 273-276.
2. Чиркин Е.С. Моделирование квантово-химическими методами кристаллической и электронной структуры молекулярных кристаллов на основе фуллеренов. / Е.С. Чиркин, Д.В. Лопатин // Вестник Тамбовского университета. Серия: Естественные и технические науки – 2009 – Т.14, №5-1 - С.887-890
3. Пузырьков Д.В. Параллельная обработка и визуализация для результатов моделирования методом молекулярной динамики. / Д. В. Пузырьков, В.О. Подрыга, С.В. Поляков // Труды ИСП РАН – 2016 – Т. 28, № 2 - С. 221-242 (на английском).
4. Жмуров, А.А. Молекулярное моделирование с использованием графических процессов. / А.А. Жмуров, В.А. Барсегов // Учебное пособие, Москва, МФТИ – 2013.
5. QuteMol [Электронный ресурс] / –Электрон. текстовые дан. –2007. –Режим доступа: <http://www.qutemol.sourceforge.net>, свободный.
6. VMD [Электронный ресурс] / –Электрон. текстовые дан. –2007. –Режим доступа: <http://www.ks.uiuc.edu/Research/vmd/>, свободный.
7. Д. Роджерс. Алгоритмические основы машинной графики: пер. с англ. – М.: Мир, 1989.
8. Никулин Е.А. Комьютерная геометрия и алгоритмы машинной графики. – СПб.: БХВ-Петербург, 2003. – 560 с.: ил.
9. Базовые алгоритмы машинной графики / С.М. Авдеева, А.В. Куров // Методические указания по выполнению лаборатоных работ по дисциплине «Машинная графика», Москва, МГТУ им. Н.Э. Баумана – 1996.
10. Филлипов, С.В. Методы и алгоритмы визуализации структурных и динамических данных, характеризующих макромолекулярные структуры. : дис. канд. физико-математических наук : 03.00.02 : автореф. дис…канд. – Пущино, 2014.
11. OpenGL ES Version 3.2 –Электрон. текстовые дан. –2007. –Режим доступа: https://www.khronos.org/registry/OpenGL/specs/es/3.2/es\_spec\_3.2.withchanges.pdf, свободный.