*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 по курсу:**

«Компьютерная графика»

**на тему:**

«Моделирование поверхности воды и обтекание водой твёрдого тела»

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Д. Н. Муслимов

(Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы \_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** М. В. Филиппов

(Подпись, дата) (И.О.Фамилия)

Москва, 2016 г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc469272264)

[1. АНАЛИТИЧЕСКИЙ РАЗДЕЛ 4](#_Toc469272265)

[1.1. Выбор алгоритма симуляции водной поверхности 4](#_Toc469272266)

[1.2. Выбор способов построения изображения. 6](#_Toc469272267)

[1.2.1. Алгоритм Робертса 7](#_Toc469272268)

[1.2.2. Алгоритм Варнока 7](#_Toc469272269)

[1.2.3. Алгоритм Z-буфера 8](#_Toc469272270)

[1.2.4. Алгоритм трассировки лучей 8](#_Toc469272271)

[1.3. Выбор алгоритма закраски 9](#_Toc469272272)

[1.3.1. Плоская закраска 9](#_Toc469272273)

[1.3.2. Метод тонирования Гуро 9](#_Toc469272274)

[1.3.3. Затенение по Фонгу 10](#_Toc469272275)

[2. КОНСТРУКТОРСКИЙ РАЗДЕЛ 12](#_Toc469272276)

[2.1. Описание алгоритма симуляции водной поверхности 12](#_Toc469272277)

[2.2. Описание модели твердого тела и сетки водной поверхности. 13](#_Toc469272278)

[2.3. Описание методов визуализации 14](#_Toc469272279)

[2.4. Аффинные преобразования 17](#_Toc469272280)

[2.5. Общая схема алгоритма визуализации сцены 18](#_Toc469272281)

[2.6. Общая схема растеризации сцены 19](#_Toc469272282)

[3. ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ 20](#_Toc469272283)

[3.1. Выбор среды разработки и языка программирования 20](#_Toc469272284)

[3.2. Диаграмма классов 21](#_Toc469272285)

[3.3. Интерфейс программы 22](#_Toc469272286)

[4. ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 25](#_Toc469272287)

[ЗАКЛЮЧЕНИЕ 27](#_Toc469272288)

[СПИСОК ЛИТЕРАТУРЫ 28](#_Toc469272289)

# ВВЕДЕНИЕ

В настоящее время перед машинной графикой стоит множество задач, большая часть из них – задачи трёхмерной графики, построение реалистичных изображений. Одна из таких задач – моделирование поверхности воды и её взаимодействия с другими телами.

В связи с этим предлагается создать программу, которая позволяет смоделировать поверхность воды, её взаимодействие с движущимся телом (лодкой) и визуализировать сцену в реальном времени.

Для решения поставленной задачи необходимо: разбив задачу на подзадачи, изучить возможные методы решения выделенных подзадач, выбрать подходящие методы и построить алгоритм их реализации, выбрать язык программирования, спроектировать интерфейс пользователя, диаграмму классов, выбрать типы и структуры данных, реализовать программу.

1. АНАЛИТИЧЕСКИЙ РАЗДЕЛ
   1. Выбор алгоритма симуляции водной поверхности

Всю сложность движущихся потоков жидкости способна описать система уравнений Навье-Стокса. В случае несжимаемой жидкости, к которой можно отнести и воду, система состоит из двух уравнений: уравнения движения и уравнения неразрывности.

Использование уравнений Навье-Стокса – это физический подход к моделированию поверхности воды. Такой подход позволяет получать реалистичную модель поверхности жидкости, но требует большого количества вычислительных ресурсов.

Существуют два основных метода решения уравнений Навье-Стокса: метод Лагранжа и метод сеток Эйлера (конечных разностей).

В методе Лагранжа жидкость рассматривается как набор частиц, подчиняющихся физическим законам. Каждая частица имеет набор параметров, таких, как масса, скорость и др. и может оказывать влияние на соседние частицы.

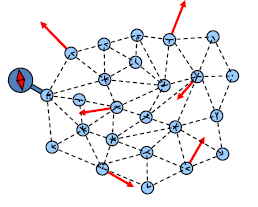


Рисунок . Взаимодействие частиц в методе Лагранжа

Методы решения уравнений, основанные на взаимодействии частиц, интуитивно понятны, их легче кодировать, к тому же, вычисление решения уравнений не требует много времени, т.к. не нужно находить решения систем уравнений. Однако, данный метод менее точен, чем метод Эйлера и не способен описать всю сложность внутреннего взаимодействия жидкости.

Высокую реалистичность визуализации жидкости может обеспечить модель гидродинамики сглаженных частиц. Главная идея данного метода – представление каждой отдельно взятой частицы как неявной поверхности, способной воздействовать на соседние частицы.

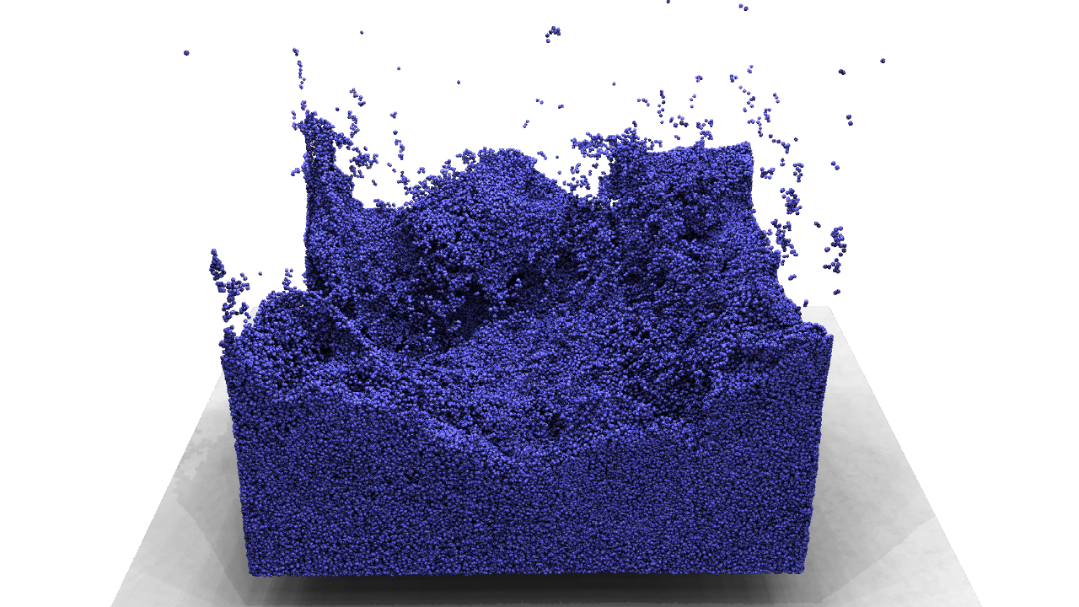


Рисунок 2. Визуализация жидкости с использованием метода гидродинамики сглаженных частиц

Тем не менее, этому методу присущи две проблемы: модель теряет свою реалистичность по мере того, как объем воды, представленной частицами, изменяется с течением времени, кроме того, для детализированной визуализации большого объема воды требуется огромное число частиц, что приводит к высоким требованиям к вычислительным ресурсам.

Метод Эйлера предполагает дискретизацию потока жидкости с помощью введения двухмерной сетки и моделирование поверхности жидкости как движение её между узлами сетки.

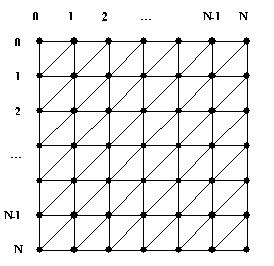


Рисунок . Общий вид равномерной сетки высот

Сеточные методы более точны, чем методы взаимодействия дискретных частиц, т.к. вычисления легче проводить с регулярной сеткой, чем с неупорядоченным набором частиц. Более того, использование регулярной сетки позволяет упростить геометрию моделируемой водной поверхности, получить более качественное изображение гладкой поверхности воды, к тому же, к координатам каждой вершины сетки легко получить доступ через двухмерный массив вершин. Однако, этот метод не лишён недостатков. Вычисления с помощью метода Эйлера будут производиться несколько дольше, чем с использованием метода Лагранжа, т.к. понадобится решение систем уравнений, с помощью сеточных методов невозможно моделировать некоторые трёхмерные свойства жидкости, например, разбивающиеся волны океана.

Вывод: хотя метод Эйлера имеет свои недостатки, в данной работе будем использовать его, т.к. он позволяет упростить геометрию, модель поверхности воды, её отображение, обладает хорошей точностью и позволяет получить реалистичную картинку.

* 1. Выбор способов построения изображения.

В настоящее время существует несколько основных алгоритмов построения реалистичных изображений: алгоритм Робертса, алгоритм Варнока, алгоритм Z-буфера, алгоритмы прямой и обратной трассировки лучей. Рассмотрим подробнее каждый из упомянутых алгоритмов.

* + 1. Алгоритм Робертса

Алгоритм Робертса – это первое известное решение задачи об удалении невидимых линий. Алгоритм выполняется в 3 этапа. На первом этапе объекты анализируются индивидуально с целью удаления нелицевых граней. На втором этапе проверяется экранирование оставшихся в каждом

теле рёбер всеми другими телами с целью обнаружения их

невидимых отрезков. На третьем этапе вычисляются отрезки, которые образуют новые рёбра при протыкании объектов друг друга.

Недостатки алгоритма:

* для работы алгоритма Робертса необходимо, чтобы все объекты сцены были выпуклыми;
* вычислительная трудоемкость алгоритма Робертса растет теоретически, как квадрат числа объектов
* невозможна реализация теней без использования посторонних методов.
  + 1. Алгоритм Варнока

В пространстве изображения рассматривается окно и решается вопрос о том, пусто ли оно, или его содержимое достаточно просто для визуализации. Если это не так, то окно разбивается на фрагменты до тех пор, пока содержимое фрагмента не станет достаточно простым для визуализации или его размер не достигнет требуемого предела разрешения. В последнем случае информация, содержащаяся в окне, усредняется, и результат изображается с одинаковой интенсивностью или цветом.

Недостаток алгоритма Варнока заключается в том, что в случае, когда невозможно сразу определить параметры закраски области, разбиение области может проводиться многократно, до тех пор, пока размер области не достигнет одного пиксела. При этом для каждой из полученных подобластей применяется процедура принятия решения. В случае отрисовки сложных сцен со множеством объектов алгоритм становится неэффективным.

* + 1. Алгоритм Z-буфера

Это один из простейших алгоритмов удаления невидимых поверхностей. Работает этот алгоритм в пространстве изображения. Идея z-буфера является простым обобщением идеи о буфере кадра. Буфер кадра используется для запоминания атрибутов (интенсивности) каждого пиксела в пространстве изображения, z-буфер - это отдельный буфер глубины, используемый для запоминания координаты z или глубины каждого видимого пиксела в пространстве изображения. В процессе работы глубина или значение z каждого нового пиксела, который нужно занести в буфер кадра, сравнивается с глубиной того пиксела, который уже занесен в z-буфер. Если это сравнение показывает, что новый пиксел расположен впереди пиксела, находящегося в буфере кадра, то новый пиксел заносится в этот буфер и, кроме того, производится корректировка z-буфера новым значением z. Если же сравнение дает противоположный результат, то никаких действий не производится. Вычислительная сложность данного алгоритма, которая равна O(n\*m\*k), где m\*n – количество пикселей в буфере кадра, k – количество полигонов.

Преимущества алгоритма Z-буфера:

* неограниченная сложность сцены;
* отсутствие необходимости нахождения пересечений объектов сцены;
* простота реализации.

Недостатки алгоритма:

* использование большого объёма памяти;
* затраты времени на отрисовку невидимых объектов.
  + 1. Алгоритм трассировки лучей

Трассировка лучей – исследование оптических систем путём отслеживания взаимодействия отдельных лучей с поверхностями. В узком смысле — технология построения изображения трёхмерных моделей в компьютерных программах, при которых отслеживается обратная траектория распространения луча.

Достоинства метода:

* возможность изображения гладких поверхностей без их аппроксимации полигонами;
* вычислительная сложность алгоритма практически не зависит от сложности сцены;
* алгоритм позволяет параллельно трассировать два луча и более, разделять участки экрана для трассирования на разных участках кластера;
* присутствует отсечение невидимых поверхностей, перспективное отображение сцены.

Главный и серьёзный недостаток метода трассировки лучей – его производительность.

Вывод: в нашей работе будем использовать алгоритм Z-буфера: несмотря на требования к памяти, данный алгоритм прост в реализации и достаточно эффективен.

* 1. Выбор алгоритма закраски
     1. Плоская закраска

Алгоритм плоской закраски довольно прост. Сперва вычисляется нормаль к текущему полигону, зачем на основе значения нормали в соответствии с выбранной моделью освещения высчитывается интенсивность цвета всего полигона. Такой метод закрашивания обладает высокой скоростью работы, но на визуализированной модели заметны переходы между гранями.

* + 1. Метод тонирования Гуро

Метод тонирования Гуро — метод закрашивания в трёхмерной компьютерной графике, предназначенный для создания иллюзии гладкой криволинейной поверхности, описанной в виде полигональной сетки с плоскими гранями, путём интерполяции цветов примыкающих граней.

Принцип метода состоит в последовательном вычислении нормалей к каждой из граней трёхмерной модели, дальнейшего определения нормалей вершин путём усреднения нормалей всех примыкающих к вершине граней. Далее на основании значений нормалей по выбранной модели отражения вычисляется освещённость каждой вершины, которая представляется интенсивностью цвета в вершине.

* + 1. Затенение по Фонгу

Затенение по Фонгу – модель расчёта освещения трёхмерных объектов, в том числе полигональных моделей и примитивов, а также метод интерполяции освещения по всему объекту.

Закраска Фонга требует больших вычислительных затрат, однако она позволяет разрешить многие проблемы метода Гуро. При закраске Гуро вдоль сканирующей строки интерполируется значение интенсивности, а при закраске Фонга - вектор нормали.

Метод Фонга используется в модели освещения для вычисления интенсивности пиксела. При этом достигается лучшая локальная аппроксимация кривизны поверхности и, следовательно, получается более реалистичное изображение. В частности, правдоподобнее выглядят зеркальные блики.

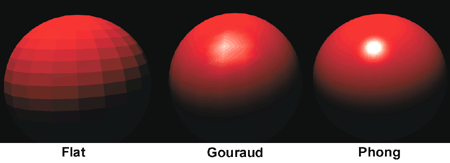


Рисунок . Сравнение методов закраски

Вывод: метод Фонга более сложен в вычислениях, чем метод Гуро, однако метод Фонга дает более реалистичное изображение, в особенности при отображении зеркальных поверхностей и бликов. Так как в данной работе необходимо визуализировать водную поверхность, было решено использовать затенение по методу Фонга.

1. КОНСТРУКТОРСКИЙ РАЗДЕЛ
   1. Описание алгоритма симуляции водной поверхности

Для физически правдоподобных способов моделирования водной поверхности и возмущений на ней, в основном, используют решения уравнений Навье-Стокса. Так как от программы, моделирующей поверхность воды, требуется работа в реальном времени, следует упростить используемую математическую модель.

Линеаризованные уравнения мелкой воды годятся для моделирования волн без учёта затухания. Уравнения мелкой воды получаются из уравнений Навье-Стокса путём интегрирования по глубине. При этом горизонтальные размеры распространяющихся по поверхности воды возмущений должен быть больше вертикального. Такие условия подходят для моделирования волн, производимых движущимся судном.

Уравнения мелкой воды:

Предполагается, что плоскость воды параллельна плоскости Oxz, u и v – скорости относительно направлений x и z соответственно, h – высота водной поверхности, d – глубина воды, g - ускорение свободного падения.

Продифференцировав первое уравнение относительно x, второе – относительно z, а третье – относительно t и объединив их в одно уравнение, получим:

Можно заметить, что при d = const полученное уравнение совпадает с волновым уравнением в двухмерном пространстве. Однако, данное уравнение не учитывает затухания волн, которое происходит в естественной среде. Уравнение можно модифицировать:

Здесь c – скорость волны, а k – параметр релаксации.

Для создания модели поверхности воды мы используем регулярную сетку:

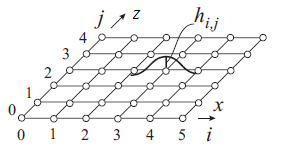


Рисунок . Равномерная сетка высот

При разностной аппроксимации получается следующий аналог предыдущей формулы для Δx = Δz:

После перегруппировки получим:

Для хранения значений высот во времена t и t – Δt будем использовать дополнительные массивы.

Воздействие движения лодки на поверхность воды реализовано за счет изменения высот полигональной сетки (уменьшение высоты узлов под лодкой, увеличение – перед лодкой).

* 1. Описание модели твердого тела и сетки водной поверхности.

Модель лодки представляет собой полигональную трехмерную модель, которая представлена в виде класса, содержащего следующие атрибуты: список координат вершин модели, список полигонов, список нормалей к вершинам, параметры материала (прозрачность, способность воспринимать фоновое, диффузное и зеркальное освещение для красного, зелёного и синего цветов).

Все полигоны модели представляют собой треугольники. Это позволяет реализовывать только один тип растеризатора.

Класс сетки поверхности воды унаследован от класса обычной модели, в модели сетки дополнительно хранится её размерность, два предыдущих по времени состояния(они необходимы для решения уравнения мелкой воды). Вершины в список вершин модели записываются последовательно, так, чтобы к ним можно было получить доступ по формуле: номер\_вершины = i\*n + j, где размерность сетки [n, m], i = 0, …, n-1; j = 0, …, m-1. Нормали к вершинам пересчитываются каждый раз после вычисления высот узлов сетки для корректной закраски.

* 1. Описание методов визуализации

Описание алгоритма z-буфера:

1. заполнить буфер кадра цветом фона, значения z установить на -∞;
2. преобразовать полигоны в растровые многоугольники в произвольном порядке;
3. для каждого пикселя (x, y) многоугольника вычислить его глубину z(x, y);
4. сравнить глубину z(x, y) с текущим значением Z(x, y) в z-буфере;
5. если z(x, y) > Z(x, y), то записать атрибуты пиксела (цвет, интенсивность) в буфер кадра, присвоить Z(x, y) = z(x, y); в противном случае не производить каких-либо действий;

Описание алгоритма затенения по Фонгу:

* + определяются нормали к граням;
  + по нормалям к граням определяются усредненные нормали в вершинах;
  + в каждой точке закрашиваемой грани определяется интерполированный вектор нормали;
  + по направлению векторов нормали определяется цвет точек грани в соответствии с принятой моделью отражения цвета.

Для нахождения вектора нормали в произвольной точке поверхности используют нормированную взвешенную сумму векторов нормали граней, которым эта точка принадлежит:



Модель освещения Фонга

В качестве модели освещения используется модель Фонга. Несмотря на то, что существую более точные модели освещения, она является стандартом в компьютерной графике.

Основная идея модели Фонга заключается в предположении, что освещенность каждой точки тела разлагается на 3 компоненты:

1. фоновое освещение (ambient);
2. рассеянный свет (diffuse);
3. бликовая составляющая (specular).

Свойства источника определяют мощность излучения для каждой из этих компонент, а свойства материала поверхности определяют её способность воспринимать каждый вид освещения.

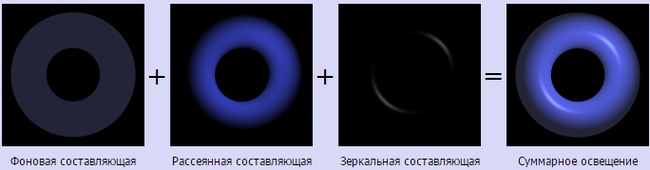


Рисунок . Модель освещения Фонга

Фоновое освещение – постоянная в каждой точке величина надбавки к освещению. Фоновая составляющая освещения вычисляется как:

, где

Ia – фоновая составляющая освещенности в точке,

ka – свойство материала воспринимать фоновое освещение,

ia – мощность фонового освещения.

Рассеянный свет при попадании на поверхность рассеивается равномерно во все стороны. При расчете рассеянной составляющей освещения учитывается только нормаль к поверхности и направление на источник света. Рассеянная составляющая рассчитывается по закону косинусов:

, где

Id – рассеянная составляющая освещенности в точке,

kd – свойство материала воспринимать рассеянное освещение,

– направление из точки на источник света,

– вектор нормали в точке,

id – мощность рассеянного освещения.

Зеркальный свет при попадании на поверхность подчиняется закону отражения: «Падающий луч, отражающий луч и перпендикуляр к границе раздела двух сред, восстановленный в точке падения луча, лежат в одной плоскости, а угол падения равен углу отражения». Отраженная составляющая освещённости в точке зависит от того, насколько близки направления на наблюдателя и отражённого луча. Интенсивность зеркальной составляющей можно вычислить по формуле:

, где

Is – зеркальная составляющая освещенности в точке,

ks – коэффициент зеркального отражения,

– направление отражённого луча,

– направление на наблюдателя,

is – мощность зеркального освещения,

β – коэффициент блеска.

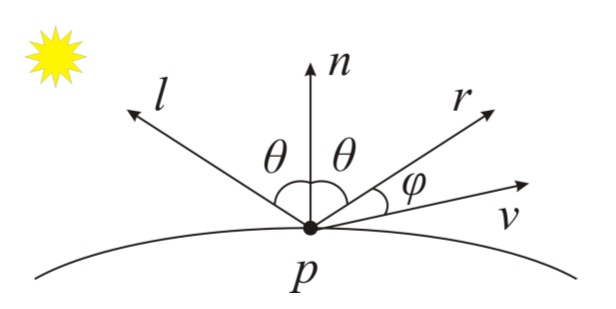


Рисунок . Отраженный свет

* 1. Аффинные преобразования

В программе реализуется возможность преобразования объектов сцены – сдвиг, поворот, масштабирование. Ниже приводятся матрицы аффинных преобразований:

* + - сдвиг точки вдоль координатных осей на dx, dy, dz:



* + - масштабирование относительно начала координат с коэффициентами kx, ky, kz:



* + - поворот относительно осей x, y, z на угол :
* вокруг оси x:



* вокруг оси y:



* вокруг оси z:



* 1. Общая схема алгоритма визуализации сцены

Загрузка модели лодки, генерация моделей воды и ландшафта

Применение возмущений, вызванных движением лодки к полигональной сетке воды

Растеризация сцены

Вычисление новых высот узлов полигональной сетки воды

Рисунок . Схема алгоритма визуализации

* 1. Общая схема растеризации сцены

Рисунок . Схема растеризации сцены

Вывод изображения на экран

Расчет освещения

Удаление невидимых поверхностей с помощью

z-буфера

Поворот сцены в соответствии с положением камеры

Проецирование

Преобразование к оконным координатам

1. ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ
   1. Выбор среды разработки и языка программирования

В качестве языка программирования был выбран язык C++. Этот язык поддерживает объектно-ориентированную парадигму программирования, что позволяет естественным образом декомпозировать задачу и легко модифицировать программу. Язык C++ содержит большой набор стандартных функций и классов, обладает высокой эффективностью.

В качестве среды был выбран Qt Creator 3.6.0. Данная среда обладает удобным редактором кода и отладчиком, широким набором настроек, вместе со средой поставляются библиотеки Qt, в том числе графические библиотеки, которые были использованы в данной работе.

Для достижения большей производительности циклы отрисовки полигонов и решения уравнений мелкой воды были распараллелены с использованием директив библиотеки OpenMP. Данная библиотека входит в состав компилятора MinGW, используемого для компиляции проекта. На однопроцессорной машине программа с использованием OpenMP может быть запущена как последовательная, нет необходимости поддерживать последовательную и параллельную версии программы.

* 1. Диаграмма классов

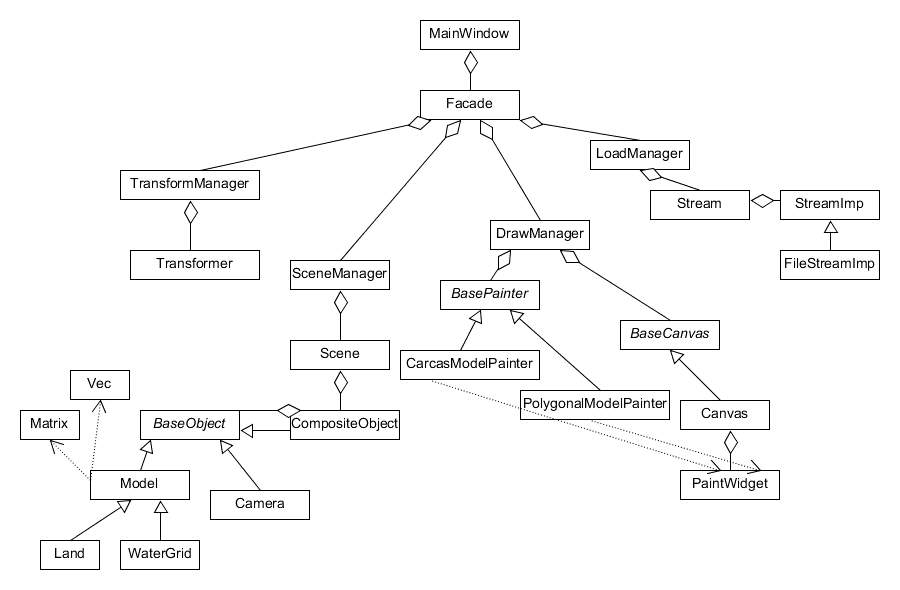


Рисунок . Диаграмма классов программы

* 1. Интерфейс программы

При запуске программы необходимо загрузить сцену в пункте меню «Опции». После этого будет построено изображение сцены.

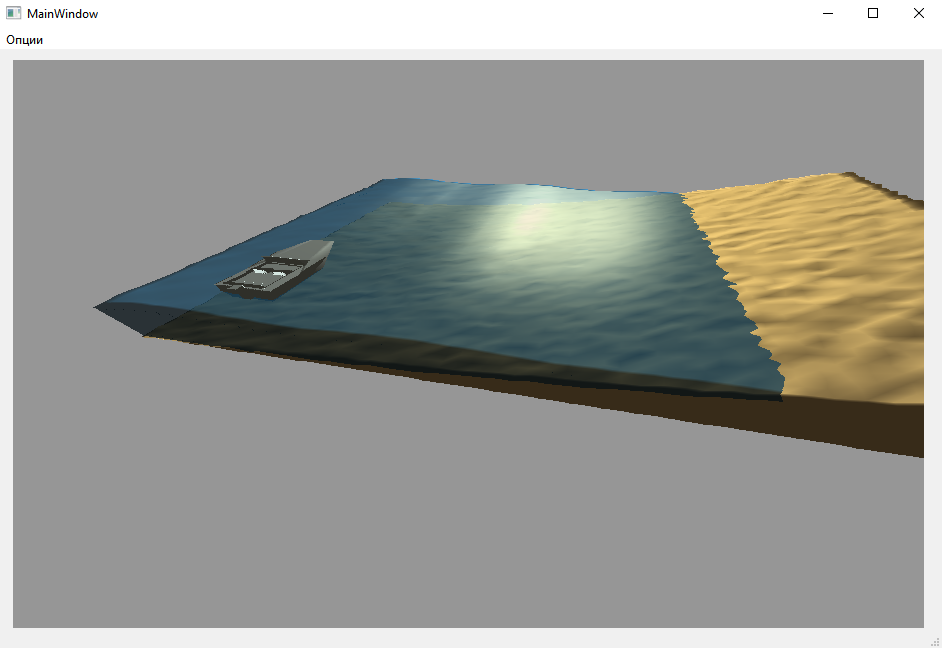


Рисунок . Интерфейс программы

Управление камерой производится с помощью клавиш: W, S –движение вперёд-назад, A, D – движение вправо-влево, Q, E – движение вниз-вверх. Лодка управляется при помощи цифровых клавиш на дополнительной клавиатуре, Num8 – движение вперёд, Num5 – движение назад, Num6 – поворот вправо, Num4 – поворот влево.

В программе предусмотрена возможность изменения положения источника света, интенсивности фоновой, диффузной и зеркальной составляющей света. Для изменения данных параметров необходимо в меню опции кликнуть на пункт «Показать окно настроек». После этого на экране появится окно настроек:



Рисунок . Настройки освещения

Примеры работы программы:

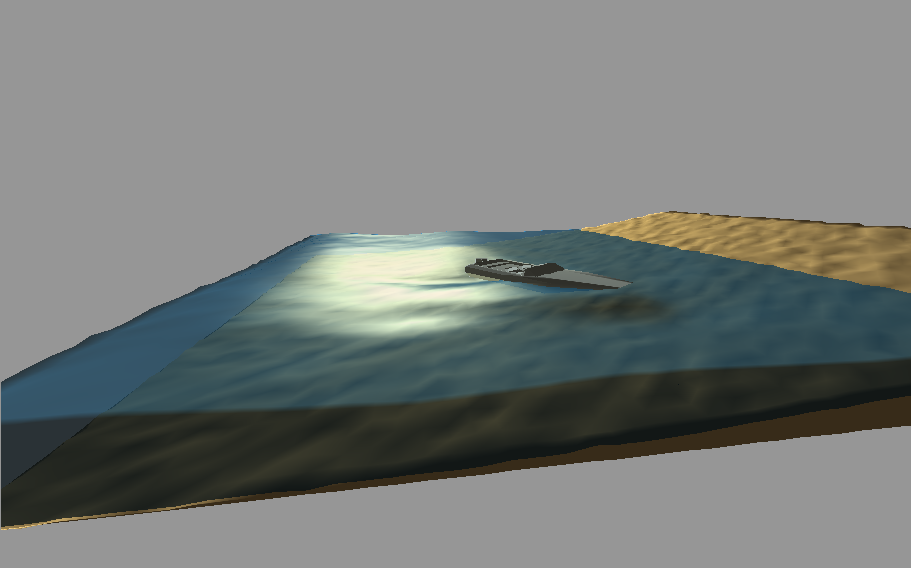


Рисунок . Пример работы

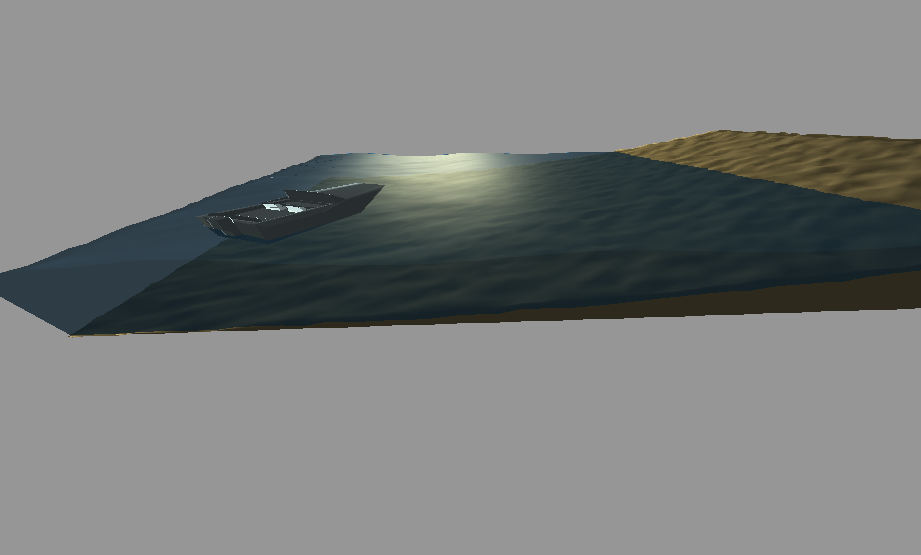


Рисунок . Пример работы

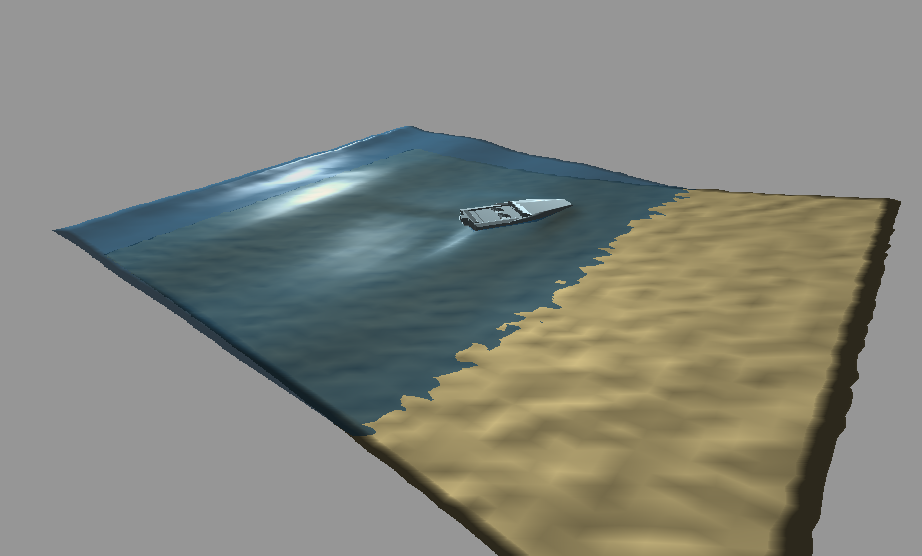


Рисунок . Пример работы

# ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

Важной характеристикой работы программы является её быстродействие. Для обеспечения более быстрой визуализации трёхмерной сцены были использованы средства распараллеливания, а именно библиотека OpenMP. В связи с этим было решено исследовать и сравнить быстродействие прямого и параллельного алгоритмов. Были произведены измерения количества тактов процессора, за которое вычислялось решение уравнений мелкой воды и визуализировалась сцена с водой (без посторонних объектов). Результаты измерений:

Рисунок . Зависимость количества тиков от количества полигонов сцены

Рисунок . Зависимость количества тиков от размерности сетки высот.

Как видно из результатов измерений, использование параллельного алгоритма позволяет сократить время визуализации сцены примерно в ~2 раза и более. Причем, чем больше количество полигонов в сцене, тем эффективнее использование распараллеленного алгоритма по сравнению с прямым.

Таким образом, использование средств распараллеливания помогает улучшить производительность программы. В качестве альтернативы распараллеливанию на процессоре и ещё большего увеличения быстродействия программы можно предложить использование архитектур параллельных вычислений на графическом процессоре (например, CUDA).

# ЗАКЛЮЧЕНИЕ

В результате проведенной работы был реализован алгоритм симуляции водной поверхности и обтекания водой твердого тела. Визуализация проведена с использованием Z-буфера и закраски Фонга с эффектом прозрачности.

В качестве перспектив развития данной программной системы можно предложить добавление в программу дополнительных объектов, взаимодействующих с водной поверхностью. Для улучшения реалистичности сцены можно добавить обработку и отрисовку теней, преломление света в воде и иные графические улучшения. Для оптимизации скорости построения изображений возможно производить визуализацию изображений на видеокарте.

# СПИСОК ЛИТЕРАТУРЫ

1. Д. Роджерс. Алгоритмические основы машинной графики./Д. Роджерс. – М.: Мир, 1989. – 512с
2. Бьерн Страуструп. Язык программирования C++. – M.: издательство «Бином», 2010. – 1136с.
3. Краткий курс компьютерной графики. [Электронный ресурс]: Хабрахабр – сайт, созданный для публикации новостей, аналитический статей и мыслей, связанных с информационными технологиями, интернетом и бизнесом. – URL: https://habrahabr.ru/post/248153/
4. Y. Nishidate and G. P. Nikishkov. Fast Water Animation Using the Wave Equation with Damping, 2005.
5. The OpenMP API specification for parallel programming. [Электронный ресурс]: The OpenMP site. – URL: http://www.openmp.org/