Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Трехмерный планировщик городского пространства

Студент: Ковалец Кирилл Эдуардович ИУ7-53Б

Научный руководитель: Вишневская Татьяна Ивановна

Цель и задачи

Целью курсовой работы является реализация программного обеспечения для трехмерного планировщика городского пространства.

Задачи:

- описать список доступных к размещению на сцене моделей, формализовать эти модели;
- выбрать существующие алгоритмы компьютерной графики для визуализации сцены и объектов на ней;
- выбрать язык программирования и среду разработки;
- реализовать выбранные алгоритмы визуализации;
- реализовать программное обеспечения для визуализации и редактирования городского пространства.

Описание и формализация объектов сцены

Объекты сцены:

- площадка сцены;
- объекты сцены:
  - дом;
  - дерево;
  - дорога;
  - машина;
- источник света.

Деревья, дома, дороги разрешено расставлять только на свободных клетках сетки сцены, машины – только на дорогах. Дороги не должны прилегать к дому.

Анализ и выбор алгоритмов

- **Алгоритмы удаления невидимых ребер и поверхностей:**
  - алгоритм Робертса;
  - алгоритм, использующий z-буфер;
  - алгоритм обратной трассировки лучей.

- **Алгоритмы построения теней:**
  - алгоритм обратной трассировки лучей;
  - алгоритм, использующий теневой z-буфер.

Алгоритм z-буфера

- Всем элементам буфера кадра присвоить фоновое значение.
- Инициализировать z-буфер минимальным значением глубины.
- Для каждого многоугольника сцены в произвольном порядке:
  - для каждого пикселя, который принадлежит многоугольнику вычислить его глубину z(x, y);
  - сравнить вычисленную глубину пикселя со значением, которое находится в z-буфере. Если z(x, y) > zbuf (x, y), то zbuf (x, y) = z(x, y) и colour(x, y) = specColour.
- Вывести итоговое изображение.

Модифицированный алгоритм z-буфера

- Для каждого направленного источника света:
  - инициализировать теневой z-буфер минимальным значением глубины;
  - определить теневой z-буфер для источника.
- Выполнить алгоритм z-буфера для точки наблюдения. При этом, если некоторая поверхность оказалась видимой относительно текущей точки наблюдения, то проверить, видима ли данная точка со стороны источников света.

  Для каждого источника света:
  - координаты рассматриваемой точки (x, y, z) линейно преобразовать из вида наблюдателя в координаты (x0, y0, z0) на виде из рассматриваемого источника света;
  - сравнить значение zshadowBuf(x0,y0) со значением z0(x0,y0). Если z0(x0, y0) < zshadowBuf (x0, y0), то пиксел высвечивается с учетом его затемнения, иначе точка высвечивается без затемнения.

Схемы алгоритмов

- Модифицированный алгоритм z-буфера
- Алгоритм z-буфера

Выбор языка программирования и среды разработки

- **При написании программного продукта использовался язык C++**:
  - данный язык преподавался в рамках курса ООП;
  - высокая вычислительная производительность;
  - язык поддерживает объектно-ориентированную парадигму программирования;
  - большое количество учебной и справочной литературы.

- **Была задействована среда разработки QT Creator:**
  - работа с QT Creator преподавалась в рамках курса Программирования на Си;
  - QT Creator позволяет работать с расширением Qt Design.

Структура классов

Пользовательский интерфейс программного обеспечения

Пример работы программного обеспечения

Пример работы программного обеспечения

Пример работы программного обеспечения

Пример работы программного обеспечения

Пример работы программного обеспечения

Проведение эксперимента

Размеры сцены	Время выполнения без использования распараллеливания, мс	Время выполнения с использованием распараллеливания, мс
1x1	25	33
10x10	152	92
20x20	203	123
30x30	227	163
40x40	251	195
50x50	296	242

Проведение эксперимента

- Распараллеливание с помощью директив библиотеки OpenMP помогло ускорить работу алгоритма z-буффера для всех рассмотренных размеров сцены, кроме сцены размером 1x1.

- Максимальная эффективность ускорения алгоритма z-буффера наблюдалась при построении сцен размерами 10x10 и 20x20 (в обоих случаях распараллеливание помогло ускорить работу алгоритма в 1.65 раз).

- Наименьшая эффективность (если не учитывать сцену размером 1x1) наблюдалась при построении сцены размером 50x50 (распараллеливание помогло ускорить работу алгоритма всего в 1.22 раза).

Заключение

В ходе выполнения курсового проекта были выполнены следующие задачи:

- описан список доступных к размещению на сцене моделей, а сами модели были формализованы;
- выбраны существующие алгоритмы компьютерной графики для визуализации сцены и объектов на ней;
- выбран язык программирования и среда разработки;
- реализованы выбранные алгоритмы визуализации;
- реализовано программное обеспечения для визуализации и редактирования городского пространства.

Поставленная цель была достигнута.