# Сопроводительная записка для презентации

# Слайд 1

Большое количество компаний, занимающихся видом деятельности, связанным с обустройством или декорированием помещений, для согласования технических заданий с клиентом часто обращаются к компьютерному графическому моделированию предмета заказа. Такой подход дает возможность как можно четче обозначить детали проекта, и реже допускать ситуации, в которых становится известно, что исполнитель неправильно воспринял мысль и идею заказчика, когда заказ уже находится на стадии завершения проектирования.

# Слайд 2

Целью моей работы является реализация программного обеспечения для визуализации площадки и интерьера выставочных стендов.

Для достижения поставленной цели потребовалось формализовать объекты синтезируемой сцены и описать список доступных к размещению моделей, выбрать и модифицировать существующие алгоритмы для визуализации сцены, реализовать данные алгоритмы и непосредственно реализовать программный продукт для визуализации и редактирования площадки выставочного стенда и трехмерных объектов, расположенных на ней.

# Слайд 3

Сцена состоит из площадки выставочного стенда, объектов интерьера и источников света. Объекты интерьера, опираясь на примеры стендов из приложения, включают в себя: стол, барную стойку, стул, барный стул, диван, растение в горшке, подиум, плазменный телевизор, шкаф и стеллаж.

# Слайд 4

Для реализации описанного программного обеспечения потребовалось проанализировать алгоритмы удаления невидимых ребер и поверхностей и алгоритмы построения теней.

Среди алгоритмов удаления невидимых ребер и поверхностей были рассмотрены: алгоритм Робертса, алгоритм, использующий z-буфер, и алгоритм обратной трассировки лучей. Из анализа достоинств и недостатков указанных алгоритмов стало ясно, что единственным из рассмотренных алгоритмов, отвечающим главному требованию – скорости работы, является алгоритм, использующий z-буфер. Недостаток этого алгоритма, кроящийся в больших затратах по памяти, решаемую проблему не затрагивает, так как синтезируемое изображение имеет достаточно малые размеры.

Среди алгоритмов построения теней были рассмотрены: алгоритм обратной трассировки лучей и алгоритм, использующий теневой z-буфер. В качестве реализуемого алгоритма была выбрана модификация алгоритма с использованием z-буфера путем добавления вычисления теневого буфера из точки наблюдения, совпадающей с источником света. Такой подход позволил не усложнять структуру программы, а также избежать проблем адаптации двух различных методов друг к другу.

# Слайд 5

Алгоритм визуализации выставочного стенда включает в себя следующие шаги: \*ссылка на слайд\*

# Слайд 6

Алгоритм z-буфера является следующей последовательностью действий: \*ссылка на слайд\*

# Слайд 7

Модифицированный алгоритм z-буфера является следующей последовательностью действий: \*ссылка на слайд\*

# Слайд 8

При написании программного продукта использовался язык C++ в связи со следующими факторами:

1) данный язык преподавался в рамках курса Объектно-Ориентированного Программирования;

2) C++ характеризуется высокой вычислительной производительностью;

3) данный язык поддерживает объектно-ориентированную парадигму программирования;

4) большое количество учебной и справочной литературы по C++ находится в свободном доступе.

Также была задействована среда разработки QT Creator в связи с тем, что:

1) основы работы с данной средой разработки преподавался в рамках курса Программирования на Си;

2) QT Creator позволяется работать с расширением Qt Design.

# Слайд 9

На текущем слайде представлены поля и методы классов точки трехмерного пространства, вершины, грани, полигональной модели и источника света.

# Слайд 10

На текущем слайде представлены поля и методы классов сцены, художника, фасада, а также классы форм пользовательского интерфейса.

# Слайд 11

Интерфейс главного окна приложения включает в себя группу работы с объектами сцены, а также группу работы со сценой. Первая группа позволяет добавлять, удалять и изменять положение добавленных на сцену объектов. Вторая группа позволяет создавать сцену и изменять её параметры.

# Слайд 12

На слайде представлены окно выбора параметров добавляемой модели и окно изменения размеров сцены.

# Слайд 13

На слайде представлены окно выбора размера сцены, окно изменения объектов сцены, окно выбора параметров источника света, а также окно задания нового положения на сцене модели.

# Слайд 14

На слайде представлен пример работы реализованного программного обеспечения с несколькими добавленными моделями и источником света с заданными углами смещения относительно текущей точки наблюдателя в 30 градусов по осям X и Y.

# Слайд 15

На данном слайде к текущей сцене был добавлен ещё один источник света с заданными углами смещения относительно текущей точки наблюдателя в 20 градусов по оси X и 30 градусов по оси Y. Таким образом, второй источник света расположен несколько ниже первого. Можно заметить, что тени несколько «сузились».

# Слайд 16

На данном слайде представлен пример работы, но уже для источника света с заданными углами смещения в 30 градусов по оси X и -30 градусов по оси Y. Теперь он находится левее точки наблюдения, тени изменились.

# Слайд 17

На данном слайде был изменён ракурс, с которого наблюдалась сцена.

# Слайд 18

Во время выполнения работы был проведен эксперимент.

Целью эксперимента было проведение оценки скорости работы программы при различных размерах визуализируемой площадки.

При реализации программного обеспечения в качестве механизма написания параллельно выполняющегося кода использовались директивы, поставляющиеся с библиотекой OpenMP. В эксперименте доказывается целесообразность использования этих директив.

Из данных, предоставленных в таблице, видно, что реализация алгоритма с использованием функционала OpenMP не показывает себя хуже обычной реализации даже на минимальных значениях.

Из графика также видно, что на всех рассмотренных размерах плоскости скорость выполнения алгоритма с использованием OpenMP больше, либо приблизительно равна скорости выполнения алгоритма без использования директив.

# Слайд 19

В среднем по таблице алгоритм с использованием директив будет быстрее реализации без их использования в 1.4 раз.

Таким образом, при конкретной реализации параллельный модифицированный алгоритм z-буфера работает быстрее обычного в любых рассмотренных условиях.