3.3 Интерфейс программы

На Рисунке. 3.1. представлен пользовательский интерфейс программы. Слева - инструментарий для редактирования сцены, справа синтезированное изображение.

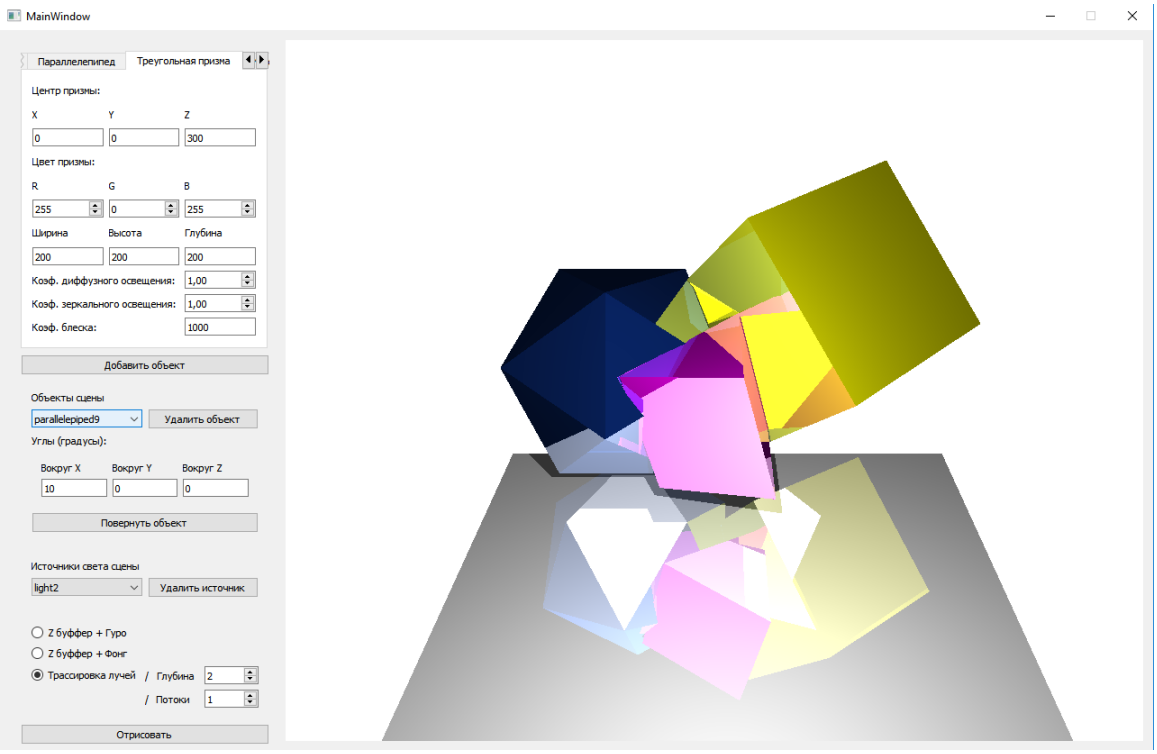

Рисунок. 3.1. Интерфейс программы

На Рисунке. 3.2. - Рисунке. 3.4. изображены окна добавления объектов на сцену. Каждое окно имеет поля для ввода:

1. центра фигуры;
2. цвета фигуры;
3. коэффициента диффузного и зеркально освещения;
4. коэффициент блеска;
5. параметры, необходымые для задания размеров объекта (Радиус для икосаэдра, Ширина, Высота и Глубина для параллелепипеда и прямой треугольной призмы)

На Рисунке. 3.5. изображено окно добавления источника света. Данное окно обладает полями для ввода координаты источника и его интенсивности.

Переключение между окнами происходит при помощи стрелок, расположенных в правом верхнем углу.

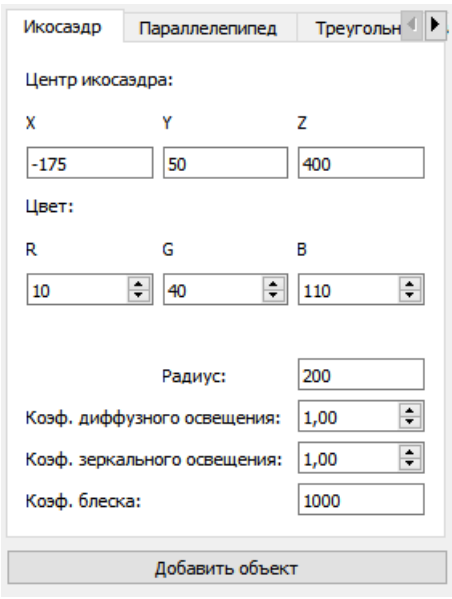

Рисунок. 3.2. Окно добавление икосаэдра

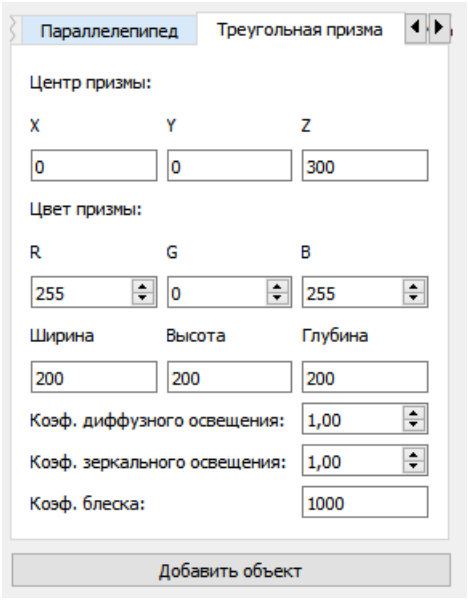

Рисунок. 3.4. Окно добавление трехгранной прямой призмы

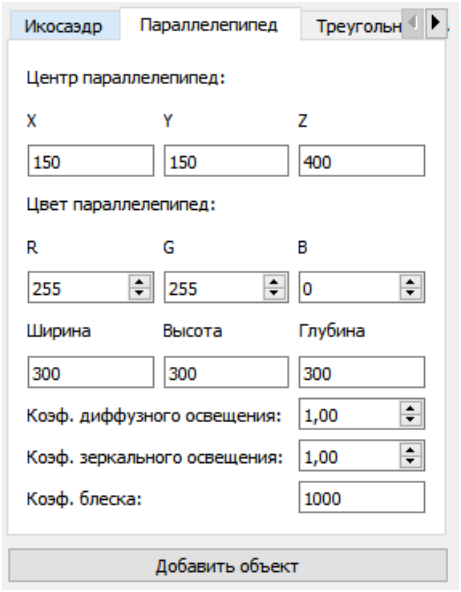

Рисунок. 3.3. Окно добавление параллелепипеда

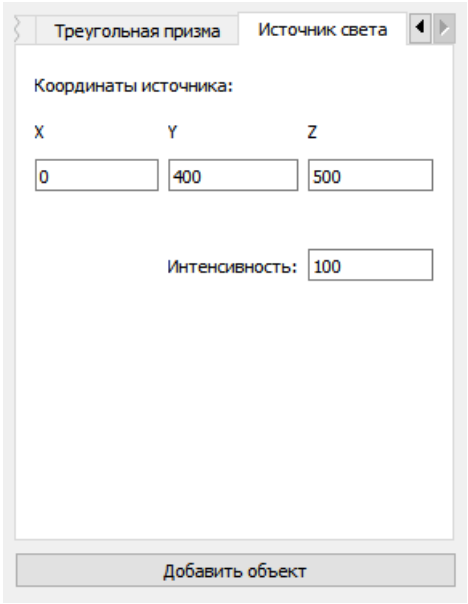

Рисунок. 3.5. Окно добавление источника света

На Рисунке. 3.6. изображены поля, предоставляющие доступ для изменения и удаления объектов, уже находящихся на сцене.

Все созданные объекты добавляются в список, далее можно выбрать объект из этого списка и удалить или повращать его.

Источники света добавляются в отдельный список, их можно только удалять.

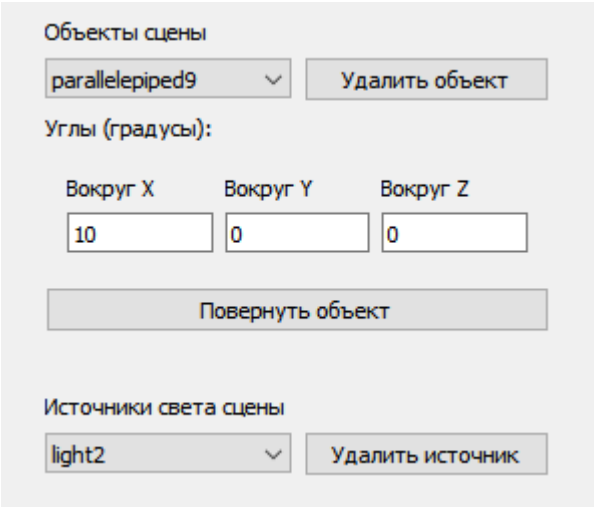

Рисунок. 3.6. Поля для изменения созданных объектов

На Рисунке. 3.7 изображены поля для выбора способа синтеза изображения.

Поле Глубина отвечает за максимальную глубину отражения при поиске отраженных лучшей для алгоритма обратной трассировки лучей.

Поле Потоки отвечает за то, на сколько потоков будет делиться процесс при синтезе изображения.

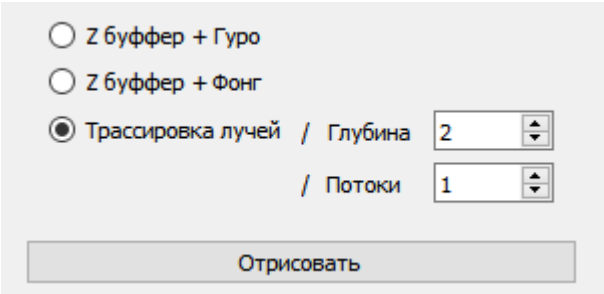

Рисунок. 3.7. Поля для выбора способо синтеза изображения

Вывод

В данном разделе обоснован выбор языка программирования, описаны выбранные средства для создания интерфейса, а так как же описано строение пользовательского интерфейса.

4 Экспериментальная часть

4.1 Цель эксперимента

Целью эксперимента является временное сравнение работы реализованных алгоритмов.

4.2 Апробация

На Рисунке 4.1. представлена сцена, содержащая куб, икосаэдр, трехгранную призму и один точечный источник. Это изображение сгенерировано обратной трассировкой лучей.

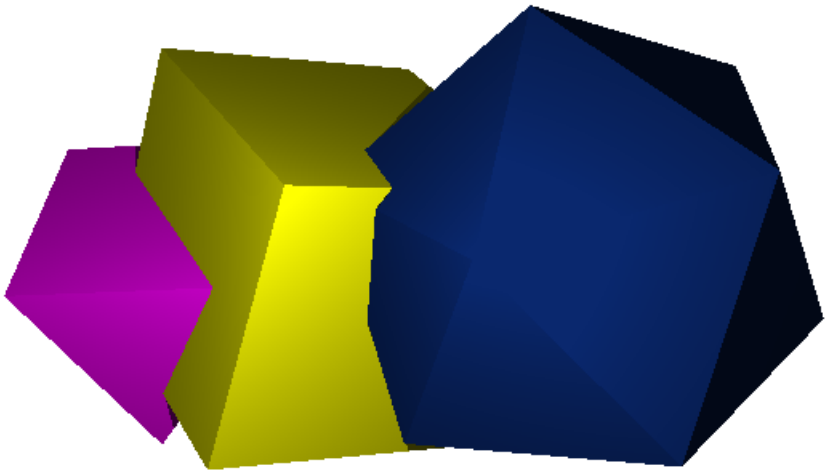

Рисунок. 4.1. Сцена, сгенерированая обратной трассировкой лучей.

На Рисунке 4.2. изображена та же самая сцена, сгенерированная обратной трассировкой, но с дополнительным вычислением отражений.

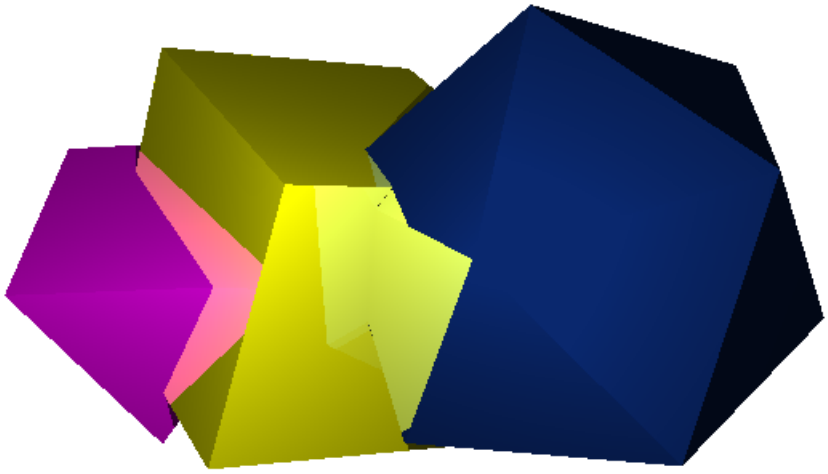

Рисунок. 4.2. Сцена, сгенерированая обратной трассировкой лучей с дополнительным вычислением отражений.

На Рисунке 4.3. показана сцена, сгенерированная с использованием Z-буфера и метода Фонга.

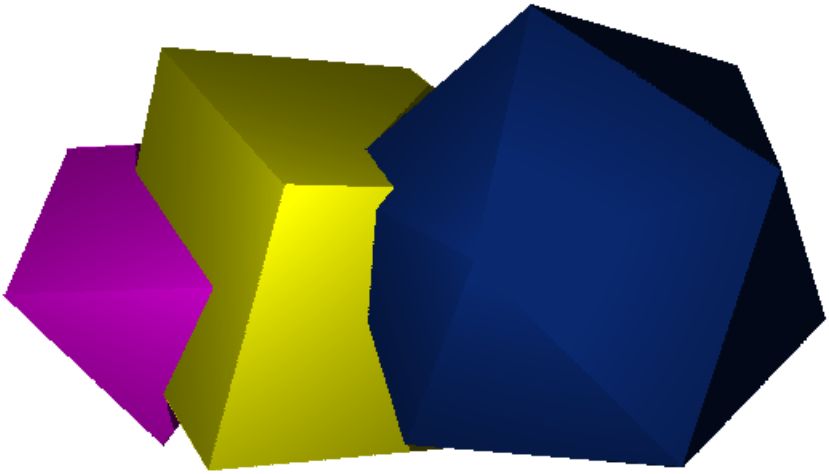

Рисунок. 4.3. Сцена, сгенерированая с использованием Z-буфера и метода Фонга.

На Рисунке 4.4. изображена сцена, сгенерированная с использованием Z-буфера и метода Гуро.

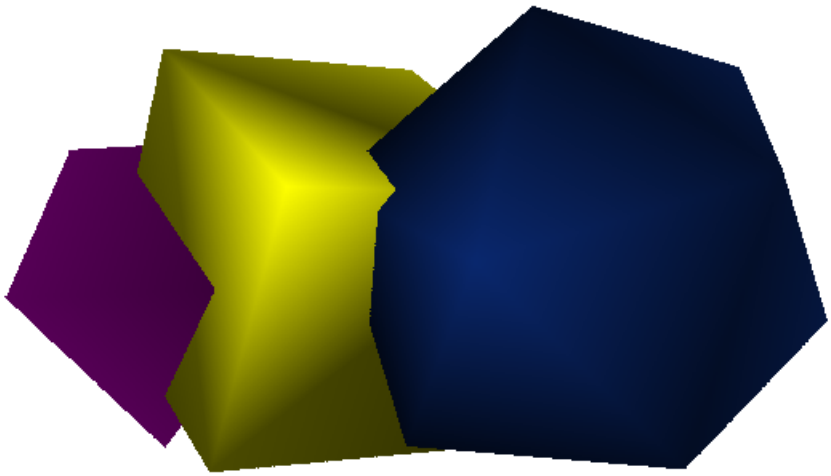

Рисунок. 4.4. Сцена, сгенерированая с использованием Z-буфера и метода Гуро.

4.3 Описание эксперимента

Суть эксперимента заключается в том, чтобы замерить скорость работы алгоритмов в двух различных случаях.

Первый случай - когда суммарное колличество граней объектов на сцене не изменяется, увеличивается лишь суммарная площадь этих граней. В теории этот случай должен быть благоприятен для использования трассировки лучей.

Второй - обратный первому, когда увеличивается суммарное колличество граней при неизменной площади. В теории этот случай должен быть благоприятен для использования Z-буфера с Фонгом или же с Гуро.

Все эксперементы проведены в однопоточном режиме.

Эксперимент проводоился на следующей системе:

1. Intel(R) Core(TM) i7-3770K
2. 8.00 ГБ ОЗУ

4.3.1 Первый случай

В данном случае на сцену добавлен один объект - парраллелипипед, содержащий 12 треугольников - граней. Во время эксперимента изменялись длины ребер, в следствии чего менялась площадь объекта.

Новых объектов не добавлялось, соответственно суммарное колличество граней не изменялось.

Таблица. 4.1. Времена работы алгоритмов при различных суммарных площадях.

Площадь (px)	Z+gourand (c)	Z+phong (c)	RayTrace (c)
160000	0.21	0.51	1.261
320000	0.261	0.555	1.271
560000	0.306	0.718	1.356
880000	0.411	0.904	1.514
1280000	0.573	1.128	1.684
2480000	1.526	2.149	1.972
3280000	2.688	2.934	2.073
3680000	3.166	3.161	2.131
4080000	3.91	3.669	2.132

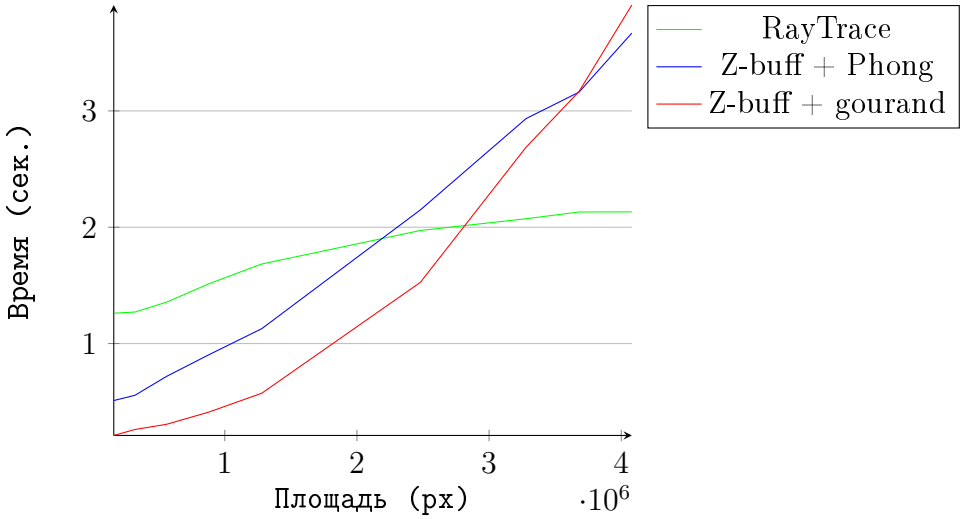

Рисунок. 4.5. График сравнения алгоритмов при различных суммарных площадях. (Таблица 4.1.)

Из Рисунка 4.5. видно, как при сильном увеличении суммарной площади без увеличения колличества граней на сцене, трассировка лучей начинает выигрывать у Z-буфера с Гуро и Z-буфера с Фонгом. (Рис. 4.5.)

Это связано с тем, что алгоритм Z-буфера переводит в растр все грани каждого объекта, соответственно при увеличении площади увеливается и трудоемкость.

В то же время обратная трассировка лучей лишь находит точку пересечения с гранью и никак не зависит от её площади.

Из этого эксперимента можно сделать вывод, что обратную трассировку лучей выгоднее использовать для синтеза сцены, на которой малое колличество граней и их суммарная площадь очень велика.

По данным Таблицы 4.1. видно, что трассировка начала работать быстрее только при суммарной площади равной 3280000 px. При таком значении трассировка быстрее алгоритмов с Z-буфером приверно в 1.45 раз. Также можно заметить, что при площаде равной 3680000 px скорость работы алгоритмов Z-буфера с закраской по Фонгу и по Гуро сравнялись, а при площаде равной 4080000 px алгоритм Z-буфера с закраской по методу Гуро начал работать на 6% медленнее Z-буфера с Фонгом.

Вывод по первому случаю: в данном случае лучшее время работы показала обратная трассировка лучей, худшее - Z-буфер с методом закраски по Гуро.

4.3.2 Второй случай

В данном случае площадь зафиксированна в значении 60000 px. На этот раз добавляются новые объекты, тем самым увеличивая суммарное колличество граней.

Таблица. 4.2. Времена работы алгоритмов при увелечении количества граней.

Кол-во граней	Z+gourand (c)	Z+phong (c)	RayTrace (c)
24	0.348	0.245	1.675
48	0.242	0.485	3.095
60	0.247	0.593	3.793
72	0.25	0.702	4.609
84	0.259	0.751	5.155

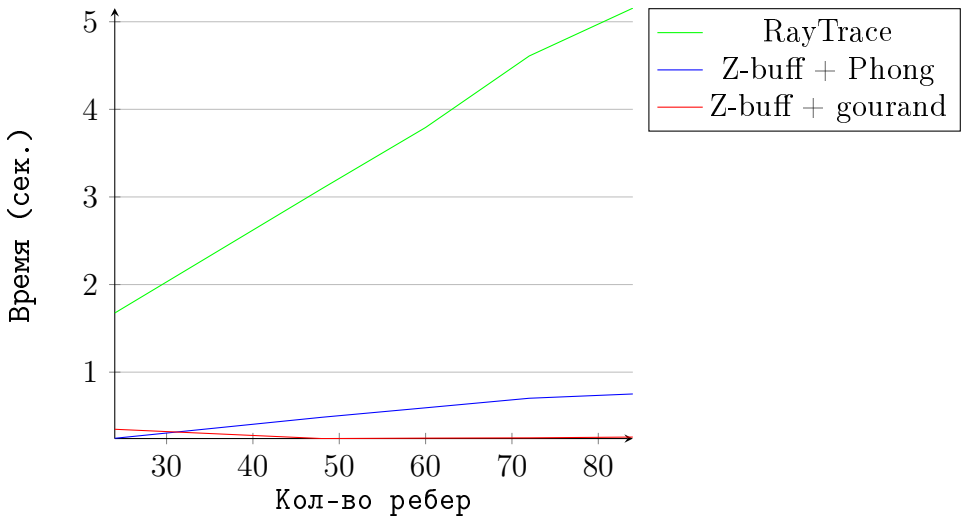

Рисунок. 4.6. График сравнения алгоритмов при увеличении количества граней.

Из Рисунка 4.6. видно, что при большом колличестве граней трассировка лучей начинает проигрывать алгоритмам с Z-буфером. Согласно Таблице 4.2., при 24 гранях трассировка работает в 6 раз медленее Z-буфера с Фонгом и в 4 раза медленее Z-буфера с Гуро. При 84 гранях разница состовляет уже 6,8 раз и 19 раз соответственно. При 84 гранях Z-буфер с Фонгом работает в 2.9 раза быстрее чем Z-буфер с Гуро.

Вывод по второму случаю: в данном случае лучшее время работы показал метод Z-буфера с закраской по Гуро, худшее - обратная трассировка лучшей.