Титульный лист

Содержание

[Введение 3](#_Toc113398429)

[1 Аналитический раздел 4](#_Toc113398430)

[1.1 Формализация объектов сцены 4](#_Toc113398431)

[1.2 Обоснование выбора формы задания трехмерных моделей 4](#_Toc113398432)

[1.2.1 Способы задания поверхностных моделей 5](#_Toc113398433)

[1.3 Обоснование выбора формы задания дыма 6](#_Toc113398434)

[1.3.1 Уравнения Навье-Стокса 7](#_Toc113398435)

[1.3.2 Система частиц 7](#_Toc113398436)

[1.4 Выбор алгоритма удаления невидимых ребер и поверхностей 9](#_Toc113398437)

[1.4.1 Алгоритм, использующий Z-буфер 10](#_Toc113398438)

[1.4.2 Алгоритм обратной трассировки лучей 10](#_Toc113398439)

[1.4.3 Алгоритм Робертса 11](#_Toc113398440)

[1.4.4 Алгоритм художника 12](#_Toc113398441)

[1.4.5 Алгоритм Варнока 13](#_Toc113398442)

[1.5 Анализ и выбор модели освещения 14](#_Toc113398443)

[1.5.1 Модель Ламберта 14](#_Toc113398444)

[1.5.2 Модель Фонга 15](#_Toc113398445)

[2 Конструкторский раздел 18](#_Toc113398446)

[2.1 Требования к программному обеспечению 18](#_Toc113398447)

[2.2 Разработка алгоритмов 18](#_Toc113398448)

[2.2.1 Общий алгоритм программы 18](#_Toc113398449)

[2.2.2 Система частиц для реализации дыма 18](#_Toc113398450)

[2.2.3 Модель освещения Ламберта 20](#_Toc113398451)

[2.3 Структура классов 20](#_Toc113398452)

[2.4 Диаграмма классов 21](#_Toc113398453)

[3 Технологический раздел 22](#_Toc113398454)

[3.1 Выбор языка программирования и среды разработки 22](#_Toc113398455)

[3.2 Интерфейс программы 22](#_Toc113398456)

[Заключение 25](#_Toc113398457)

Введение

В современном мире компьютерная графика находит свое применение в самых различных областях. Типичный пример – это кино и игры.

Алгоритмы получения реалистичных изображений на сегодняшний день получают очень много внимания. Эти алгоритмы сложны в реализации и являются одними из самых затратных по ресурсам. Они должны учитывать не только формы и текстуры объектов, но и множество других физических явлений, таких как преломление, отражение, рассеивание света.

Очевидно, что более получение качественного изображения на выходе алгоритма, требует больше времени и памяти для синтеза. Хотя для генерации статических изображений это не является проблемой, при создании динамической сцены могут возникать большие сложности, так как каждый раз на очередном временном интервале придется рассчитывать все заново.

Практика проводилась на Истоке в отдел автоматизированных систем управления в секторе разработке и сопровождения ПО. Давалась консультация по разным спорным вопросам.

Целью данной практики является подготовка к выполнению курсовой работы, в которой будет реализована визуализация такого явления как распространение света в неравномерно задымленном помещении. Явление будет продемонстрировано на примере комнаты, в которой находятся источник дыма и источник света.

Для достижения поставленной цели, мне потребуется решить следующие задачи:

* описать модель трехмерной сцены, в том числе и объекты, из которых она состоит;
* проанализировать существующие алгоритмы построения изображения и выбрать из них те, что в лучше помогут в решении поставленной задачи.

1. Аналитический раздел

В данном разделе представлено описание объектов сцены, а также обоснован выбор алгоритмов, которые будут использованы для ее визуализации.

* 1. Формализация объектов сцены

Сцена состоит из следующих объектов:

* источников света:
* источник направленного света – задается вектором направления, интенсивностью и положением в пространстве;
* точечный источник света - задается положением в пространстве и интенсивностью;
* источника дыма – задается положением в пространстве;
* модели дыма;
* комнаты – пространство в пределах, которого будет распространяться дым;
* предметов интерьера – объекты внутри комнаты, мешающие распространению дыма и света.
  1. Обоснование выбора формы задания трехмерных моделей

Отображением формы и размеров объектов являются модели. Обычно используются три формы задания моделей.

1. Каркасная (проволочная) модель.

Одна из простейших форм задания модели, так как мы храним информацию только о вершинах и ребрах нашего объекта.

1. Поверхностная модель.

Поверхностная модель объекта — это оболочка объекта, пустая внутри. Такая информационная модель содержит данные только о внешних геометрических параметрах объекта. При этом могут использоваться различные типы поверхностей, ограничивающих объект, такие как полигональные модели, поверхности второго порядка и др.

1. Объемная (твердотельная) модель.

При твердотельном моделировании учитывается еще материал, из которого изготовлен объект, т.е. у нас есть информация о том, с какой стороны поверхности расположен материал.

Для решения данной задачи лучше всего подойдет поверхностная модель. Каркасные модели могут привести к неправильному восприятию формы объекта, а реализация объемной модели избыточна и потребует дополнительного количества ресурсов.

* + 1. Способы задания поверхностных моделей

На следующем шаге необходимо определиться со способом задания поверхностной модели.

1. Аналитический способ.

Этот способ задания модели характеризуется описанием модели объекта, которое доступно в неявной форме, то есть для получения визуальных характеристик необходимо дополнительно вычислять некоторую функцию, которая зависит от параметра.

1. Полигональная сетка.

Данный способ характеризуется совокупностью вершин, граней и ребер, которые определяют форму многогранного объекта в трехмерной компьютерной графике.

При этом существует несколько способов хранения информации о сетке.

1. Список граней.

Объект – это множество граней и множество вершин. В каждую грань

входят как минимум 3 вершины;

1. «Крылатое» представление.

Каждая точка ребра указывает на две вершины, две грани и четыре ребра, которые её касаются.

1. Полурёберные сетки.

То же «крылатое» представление, но информация обхода хранится

для половины грани.

1. Таблица углов.

Это таблица, хранящая вершины. Обход заданной таблицы неявно

задаёт полигоны. Такое представление более компактно и более производительно для нахождения полигонов, но, в связи с тем, что вершины присутствуют в описании нескольких углов, операции по их изменению медленны.

1. Вершинное представление.

Хранятся лишь вершины, которые указывают на другие вершины.

Простота представления даёт возможность проводить над сеткой множество операций.

Вывод

Одним из важнейших факторов в выборе способа реализации модели является скорость вычисления преобразований над объектом. Поэтому для задания модели наилучшим представлением будет полигональная сетка. При этом способ хранения полигональной сетки – список граней. Такое представление является наиболее простым, но в то же время наиболее удобным для решения поставленной задачи. Также этот метод дает возможность быстрого преобразования модели, потому что структура будет включать в себя список вершин.

* 1. Обоснование выбора формы задания дыма

Дым представляет собой дисперсную систему, состоящую из твердых частиц, находящихся во взвешенном состоянии. За всю историю моделирования было разработано несколько методов, подробнее о которых я расскажу дальше.

* + 1. Уравнения Навье-Стокса

Поведение частиц дыма аналогично поведению легких газов. Частицы дыма легкие и мелкие, и при испускании из неподвижного источника, такого как свеча, не будут иметь скорости. Движение частиц в основном вызывают окружающие силы. Частицы дыма не влияют на свое собственное движение, а зависят исключительно от внешних сил, толкающих и тянущих их в разных направлениях. Дым, как правило, вытягивается вверх из-за малого веса частиц, которые следуют за теплым воздушным потоком, поднимающимся вверх. Дым, как и жидкости, также будет следовать за поверхностями при взаимодействии с объектами. Для моделирования такого поведения часто используется вычислительная гидродинамика (Computational Fluid Dynamics). Обычно используемыми уравнениями для решения CFD являются уравнения Навье-Стокса и Эйлера. Оба уравнения описывают, как изменяются скорости и массы и предполагая, что вязкость равна нулю, удаление теплопроводности из уравнения Навье-Стокса дает уравнение Эйлера.

– плотность,

*p* – давление,

*f* – произвольная сила,

*u* – скорость частицы.

* + 1. Система частиц

Движение частиц

Основное движение дыма происходит вверх по спирали. Очевидно, что для достижения более хаотичное движение, потребуется добавление 3D-шума. Эти две составляющие движения рассчитываются отдельно. Спираль, по которой движутся частицы, управляется простым векторным полем.

(1)

где *xpos, ypos* и *zpos* - мировые координаты частицы, а *yvel* - скорость, с которой частицы движутся вверх.

Это значение одинаково везде в пространстве. Сила тяжести не оказывает значительное влияние на частицы дыма и не принимается во внимание. Также не происходит столкновений между частицами дыма.

Частицам присваиваются скорости в соответствии с их положением в векторном поле. Затем их новые позиции вычисляются с использованием интегрирования Эйлера первого порядка:

(2)

где p(t) - положение частицы в момент времени t, а v - трехмерный вектор, представляющий скорость частицы, заданную векторным полем.

Конечно, это не конечное положение частицы. Шум добавляется для создания более хаотичного движения. Все частицы рождаются со скоростью (0,1,0). Частицы перемещаются в соответствии с их текущей скоростью, прежде чем им присваивается новая скорость, которую они сохраняют до следующего кадра.

Шум частиц

Шум, используемый для частиц, отличается от обычного шума главным образом возвращаемым из него значением. Обычный шум возвращает значение от нуля до единицы, тогда как шум, используемый для системы частиц, возвращает 3D-вектор.

Затухание частиц

Густой, теплый дым вначале имеет тенденцию быть темным и прозрачным, а по мере продвижения вверх становится все гуще. Противоположное можно сказать о более тонком, более похожем на пар дыме.

Затухание выполняется с использованием альфа-значений и основано на возрасте частиц.

Возраст частиц колеблется от нуля до максимального. В случае густого, теплого дыма нулевой возраст частиц соответствует нулевому альфа-значению, а максимально допустимый возраст соответствует альфа-значению, равному единице. Это означает, что с возрастом частицы становятся более непрозрачными. Противоположное верно для пароподобного дыма, частицы которого изначально непрозрачны и исчезают с возрастом.

Время жизни частицы

Все частицы абсолютно идентичны при рождении. В промежутках между кадрами сохраняются положение, скорость и возраст частиц. В начале каждого кадра рождается фиксированное количество частиц. Частицы удаляются, когда они превысили максимальный возраст частиц, который является глобальным и, следовательно, одинаковым для всех частиц.

Вывод

Хотя CFD метод даст более реалистичное изображение, он очень сложен в реализации и требует большой вычислительной мощности. Из-за таких серьезных недостатков для выполнения проекта будет выбран метод, основанный на частицах.

* 1. Выбор алгоритма удаления невидимых ребер и поверхностей

Выбранный алгоритм удаления невидимых ребер должен обладать некоторым радом свойств, а именно:

* из-за большого количества частиц, алгоритм должен использовать минимум памяти;
* алгоритм должен быть достаточно быстрым;
* как результат работы алгоритм должен выдавать максимально реалистичное изображение.
  + 1. Алгоритм, использующий Z-буфер

Суть данного алгоритма заключается в использовании двух буферов: буфера кадра, в котором хранятся атрибуты каждого пикселя, и Z-буфера, в котором хранится информация о координате Z для каждого пикселя.

Первоначально в Z-буфере находятся минимально возможные значения Z, а в буфере кадра располагаются пиксели, описывающие фон. Каждый многоугольник преобразуется в растровую форму и записывается в буфер кадра.

В процессе подсчета глубины нового пикселя, он сравнивается с тем значением, которое уже лежит в Z-буфере. Если новый пиксель расположен ближе к наблюдателю, чем предыдущий, то он заносится в буфер кадра и происходит корректировка Z-буфера.

Для решения задачи вычисления глубины Z каждый многоугольник описывается уравнением . При многоугольник для наблюдателя вырождается в линию.

Для некоторой сканирующей строки , поэтому имеется возможность рекуррентно высчитывать для каждого : . Получим , так как 𝑥 − 𝑥 ′ = 𝑑𝑥 = 1. При этом стоит отметить, что для невыпуклых многогранников предварительно потребуется удалить не лицевые грани.

Преимущества

* простота реализации;
* оценка трудоемкости линейна.

Недостатки

* сложная реализация прозрачности;
* большой объем требуемой памяти.
  + 1. Алгоритм обратной трассировки лучей

Алгоритмы трассировки лучей на сегодняшний день считаются наиболее мощными при создании реалистичных изображений. Изображение формируется из-за того, что свет попадает в камеру.

Выпустим из источников света множество лучей (первичные лучи). Часть этих лучей не встретит никаких препятствий, а часть попадет на объекты. На них лучи могут преломляться и отражаться. При этом часть энергии луча поглотится. Преломленные и отраженные лучи образуют новое поколение лучей. Далее эти лучи опять же преломятся, отразятся и образуют новое поколение лучей. В конечном итоге часть лучей попадет в камеру и сформирует изображение. Это описывает работу прямой трассировки лучей.

Метод обратной трассировки лучей позволяет значительно сократить перебор световых лучей. В этом методе отслеживаются лучи не от источников, а из камеры. Таким образом, трассируется определенное число лучей, равное разрешению картинки.

Преимущества

* высокая реалистичность синтезируемого изображения;
* работа с поверхностями в математической форме;
* вычислительная сложность слабо зависит от сложности сцены.

Недостатки

* производительность.
  + 1. Алгоритм Робертса

Данный алгоритм работает в объектном пространстве, решая задачу только с выпуклыми телами. Алгоритм выполняется в 3 этапа.

Этап подготовки исходных данных.

На данном этапе должна быть задана информация о телах. Для каждого тела сцены должна быть сформирована матрица тела V. Размерность матрицы - 4 \* 𝑛, где 𝑛 – количество граней тела.

Каждый столбец матрицы представляет собой четыре коэффициента уравнения плоскости 𝑎𝑥 + 𝑏𝑦 + 𝑐𝑧 + 𝑑 = 0, проходящей через очередную грань.

Матрица тела должна быть сформирована корректно, то есть любая точка, расположенная внутри тела, должна располагаться по положительную сторону от каждой грани тела. В случае, если для очередной грани условие не выполняется, соответствующий столбец матрицы надо умножить на -1.

Этап удаления рёбер, экранируемых самим телом.

На данном этапе рассматривается вектор взгляда 𝐸 = {0, 0, − 1, 0}. Для определения невидимых граней достаточно умножить вектор 𝐸 на матрицу тела 𝑉. Отрицательные компоненты полученного вектора будут соответствовать невидимым граням.

Этап удаления невидимых рёбер, экранируемых другими телами сцены.

На данном этапе для определения невидимых точек ребра требуется построить луч, соединяющий точку наблюдения с точкой на ребре. Точка будет невидимой, если луч на своём пути встречает в качестве преграды рассматриваемое тело.

Преимущества

* работа в объектном пространстве;
* высокая точность вычисления.

Недостатки

* рост сложности алгоритма – квадрат числа объектов;
* тела сцены должны быть выпуклыми (усложнение алгоритма, так как нужна будет проверка на выпуклость);
* сложность реализации.
  + 1. Алгоритм художника

Предназначен для изображения произвольных поверхностей. Принцип его работы заключается в выводе на экран объектов по мере их приближения к наблюдателю. Наиболее распространенная реализация алгоритма – сортировка по глубине, которая заключается в том, что произвольное множество граней сортируется по ближнему расстоянию от наблюдателя, а затем отсортированные грани выводятся на экран в порядке от самой дальней до самой ближней.

Данный метод работает лучше для построения сцен, в которых отсутствуют пересекающиеся грани.

Преимущества

* требование меньшей памяти, чем, например, алгоритм Z-буфера.

Недостатки

* недостаточно высокая реалистичность изображения;
* сложность реализации при пересечения граней на сцене.
  + 1. Алгоритм Варнока

Алгоритм Варнока является одним из примеров алгоритма, основанного на разбиении картинной плоскости на части, для каждой из которых исходная задача может быть решена достаточно просто. Поскольку алгоритм Варнока нацелен на обработку картинки, он работает в пространстве изображения.

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. Сравнивая область с проекциями всех граней, можно выделить случаи, когда изображение, получающееся в рассматриваемой области, определяется сразу:

* проекция ни одной грани не попадает в область;
* проекция только одной грани содержится в области или пересекает область, то в этом случае проекции грани разбивают всю область на две части, одна из которых соответствует этой проекции;
* существует грань, проекция которой полностью накрывает данную область, и эта грань расположена к картинной плоскости ближе, чем все остальные грани, проекции которых пересекают данную область, то в данном случае область соответствует этой грани.

Если ни один из рассмотренных трех случаев не имеет места, то снова разбиваем область на четыре равные части и проверяем выполнение этих условий для каждой из частей. Те части, для которых таким образом не удалось установить видимость, разбиваем снова и т. д.

Преимущества

* меньшие затраты по времени в случае области, содержащий мало информации.

Недостатки

* алгоритм работает только в пространстве изображений;
* большие затраты по времени в случае области с высоким информационным содержимым.

Вывод

Для удаления невидимых линий выбран алгоритм обратной трассировки лучей. Этот алгоритм не только поможет добиться наибольшей реалистичности изображения, но и также позволит легко смоделировать распространение света в пространстве, согласно законам геометрической оптики. Данный алгоритм можно улучшить, если добавить в него обработку особых световых явлений. Помимо всего прочего данный алгоритм позволит построить качественные тени. Также немаловажен тот факт, что алгоритм трассировки лучей намного менее требователен к памяти, чем тот же алгоритм Z-буфера.

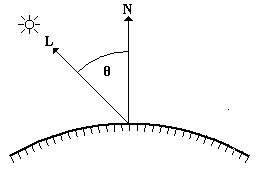
* 1. Анализ и выбор модели освещения

Физические модели материалов стараются аппроксимировать свойства некоторого реального материала. Такие модели учитывают особенности поверхности материала или же поведение частиц материала. Эмпирические модели материалов устроены иначе, чем физически обоснованные. Данные модели подразумевают некий набор параметров, которые не имеют физической интерпретации, но которые позволяют с помощью подбора получить нужный вид модели. В данной работе следует делать выбор из эмпирических моделей, а конкретно из модели Ламберта и модели Фонга.

* + 1. Модель Ламберта

Модель Ламберта моделирует идеальное диффузное освещение, то есть свет при попадании на поверхность рассеивается равномерно во все стороны.

Интенсивность диффузного света приблизительно подчиняется закону Ламберта, то есть интенсивность пропорциональна только косинусу угла между направлением падающего света и нормалью к поверхности в точке отражения.



*Рисунок 1.1*

(3)

(4)

(5)

Эта модель является одной из самых простых моделей освещения и очень часто используется в комбинации с другими моделями. Она может быть очень удобна для анализа свойств других моделей, за счет того, что ее легко выделить из любой модели и анализировать оставшиеся составляющие.

* + 1. Модель Фонга

Это классическая модель освещения. Модель представляет собой комбинацию диффузной и зеркальной составляющих. Работает модель таким образом, что кроме равномерного освещения на материале могут появляться блики. Местонахождение блика на объекте определяется из закона равенства углов падения и отражения. Чем ближе наблюдатель к углам отражения, тем выше яркость соответствующей точки. Падающий и отраженный лучи лежат в одной плоскости с нормалью к отражающей поверхности в точке падения

где – коэффициент фонового освещения

– коэффициент рассеянного освещения

– коэффициент бликового освещения

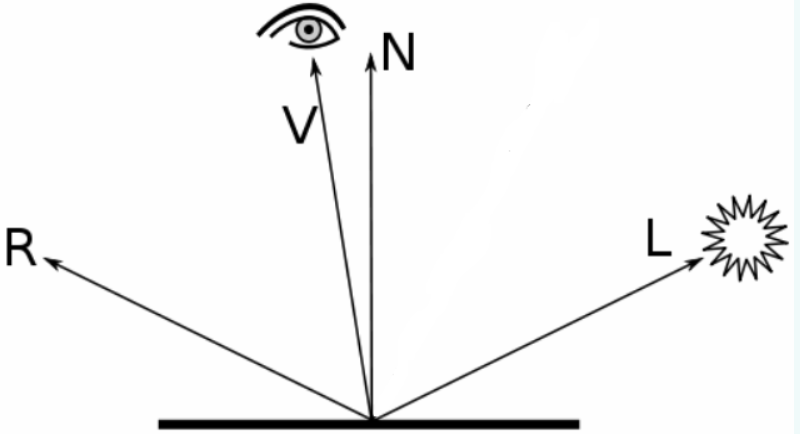
– вектор нормали к поверхности

– вектор направления падающего луча

– вектор направления отраженного луча

– вектор взгляда

– контрастность блика.



*Рисунок 1.2*

Вывод

В качестве модели освещения в данной работе была выбрана модель Ламберта из-за своей простоты по сравнению с моделью Фонга. На расчет данных для модели Ламберта потребуется выполнить меньше вычислений, а значит ее реализация будет проще и потребует меньшего количества времени.

Вывод

В данном разделе был проведен анализ алгоритмов удаления невидимых линий и модели освещения, которые возможно использовать для решения поставленных задач, а также выбраны формы задания дыма и объектов на сцене. В качестве ключевого алгоритма, выбран алгоритм обратной трассировки лучей, который будет реализован в рамках курсового проекта по компьютерной графике.

1. Конструкторский раздел

В данном разделе будут представлены требования к программному обеспечению, а также схемы алгоритмов, выбранных для решения задачи.

* 1. Требования к программному обеспечению

Программа должна предоставлять доступ к функционалу:

* изменение параметров источника дыма в активном режиме: интенсивность генерации частиц дыма, расположение в комнате, начальная скорость вылета частиц;
* изменение параметров частиц, из которых состоит дым, в активном режиме: количество частиц, размер частиц;
* запуск и постановка на паузу сцены;
* вращение, перемещение и масштабирование модели.

Требования, которые предъявляются к программе:

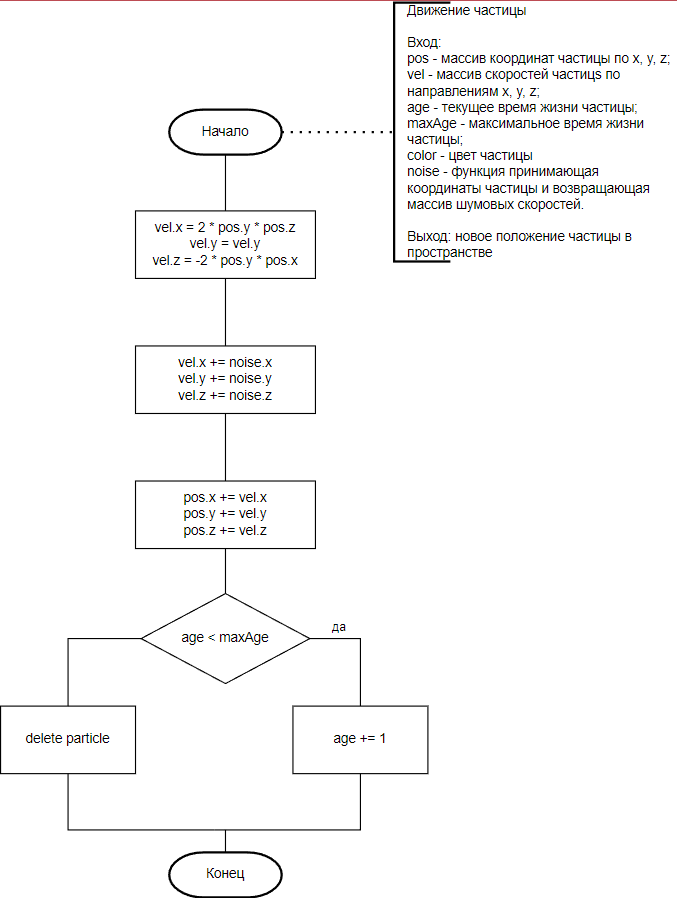
* время отклика программы должно быть менее 1 секунды для корректной работы в интерактивном режиме;
* программа должна корректно реагировать на любое действие пользователя.
  1. Разработка алгоритмов

В данном разделе будут представлены схемы реализации выбранных алгоритмов.

* + 1. Общий алгоритм программы

1. Задать объекты сцены (интерьер комнаты, вид и расположение источников сцены, характеристики источников дыма).
2. Задать параметры частиц дыма.
3. Запустить симуляцию.
4. По требованию пользователя остановить симуляцию.
   * 1. Система частиц для реализации дыма

На рисунке 2.1 показана схема алгоритма реализации движения частицы.



*Рисунок 2.1 Схема алгоритма движения частиц*

* + 1. Модель освещения Ламберта

Этот способ освещения вычисляет цвет поверхности в зависимости от того по каким углом падает свет. Интенсивность света вычисляется по формуле.

(6)

где I – интенсивность источника света,

L – вектор обратный направлению луча света,

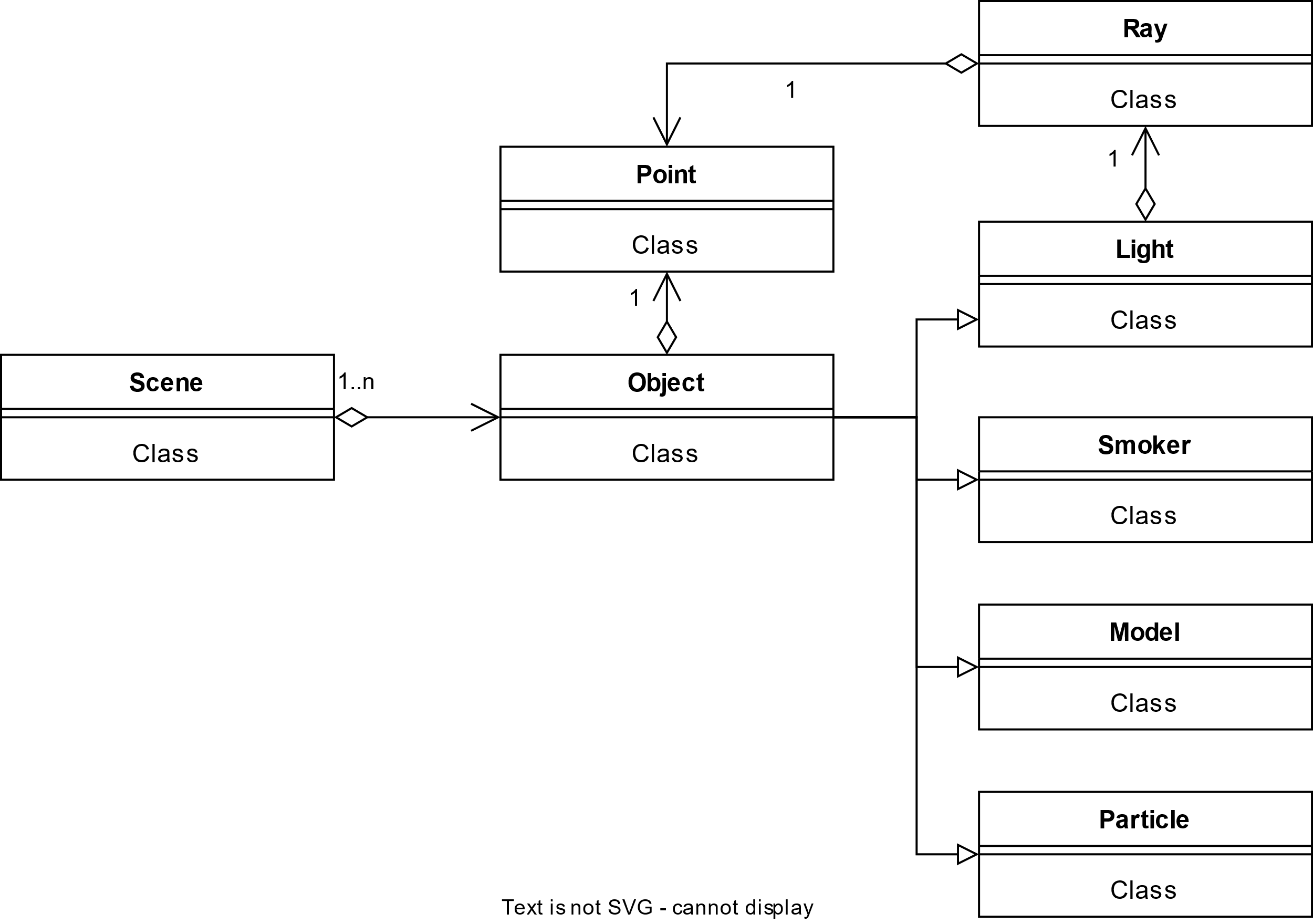
N – вектор нормали.

* 1. Структура классов

Классы в программе можно условно разделить на несколько групп по выполняемым функциям.

* Математические абстракции
  + Ray - трехмерный луч, задающийся точкой начала луча, направляющим вектором;
  + Point – радиус-вектор точки сцены.
* Трехмерные объекты
  + Model – класс хранящий в себе статический декоративный объект;
  + Light – класс источника света, содержащий информацию о направлении положении и интенсивности;
  + Smoker – класс источника дыма, содержащий информацию о скорости генерации частиц и положении.
  + Particle – класс частицы дыма, содержащий информацию о положении, скорости, времени жизни, размерах, цвете частицы.
* Сцена
  + Scene - характеризует набор объектов и их свойств.
* Интерфейс пользователя
  + Взаимодействие с интерфейсом происходит через диалоговые окна, которые в свою очередь взаимодействуют с классом Scene.
  1. Диаграмма классов

На рисунке 2.2 представлена диаграмма классов



*Рисунок 2.2 – диаграмма классов.*

Вывод

В данном разделе были рассмотрены требования, которые выдвигаются программному продукту, схемы алгоритмов, а также типы и структуры данных, которые были использованы при реализации ПО.

1. Технологический раздел
   1. Выбор языка программирования и среды разработки

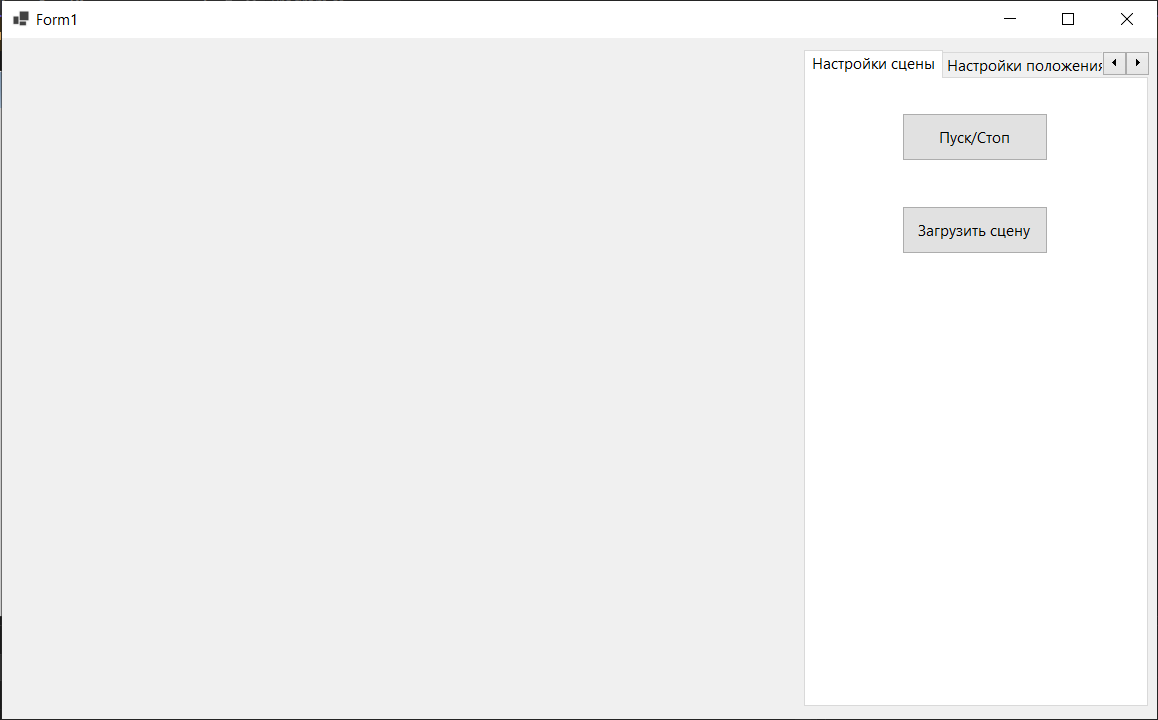
Для разработки данной программы был выбран язык C#. Данный выбор обусловлен следующими факторами:

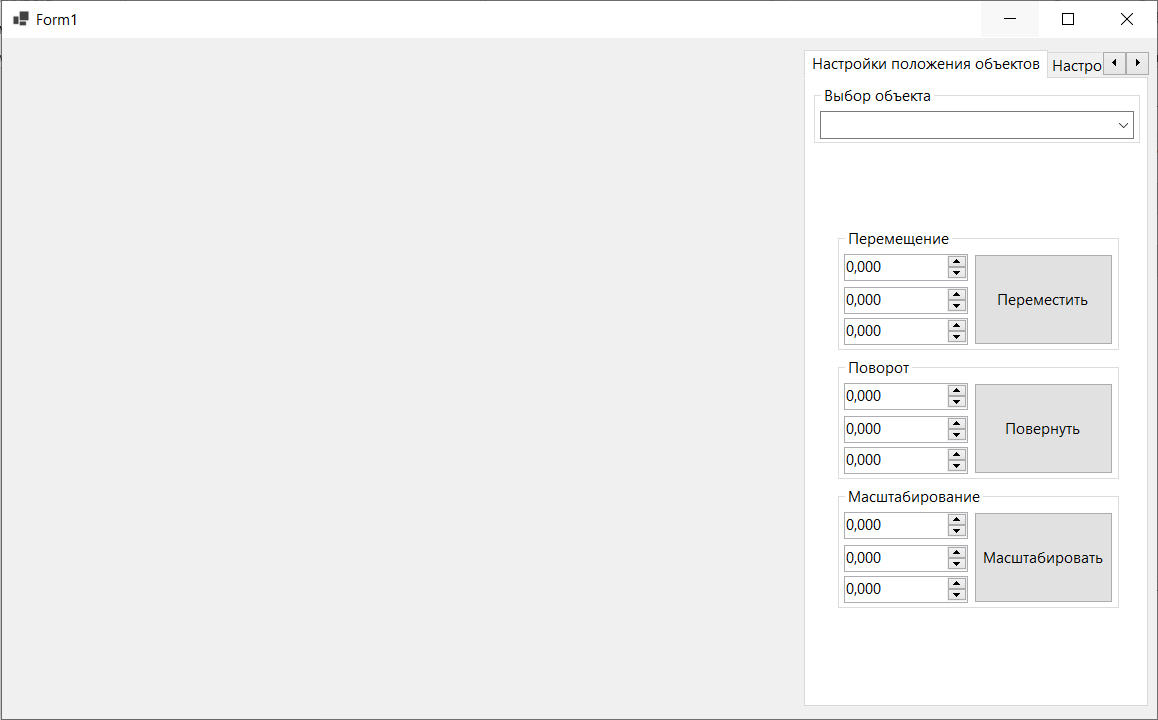
* Язык предоставляет программисту широкие возможности реализации самых разнообразных алгоритмов. Он обладает высокой эффективностью и большим набором стандартных классов и процедур.
* Язык C# является полностью объектно-ориентированным.
* Все необходимые библиотеки для реализации поставленной задачи являются стандартными.

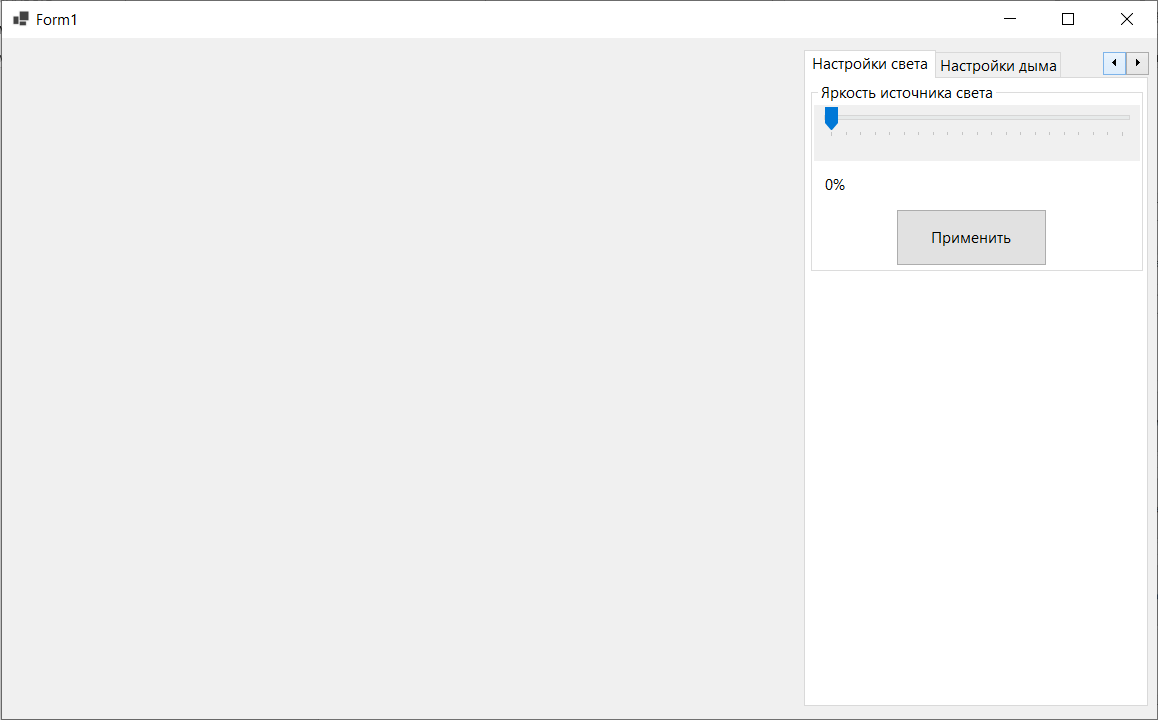
В качестве среды разработки была выбрана Visual Studio 2022. Среда позволяет работать с .NET Framework что предоставляет широкий функционал, а именно:

* Мощная библиотека классов. .NET представляет единую для всех поддерживаемых языков библиотеку классов.
* Автоматическая сборка мусора, как особенность языка C# и фреймворка .NET.
* Работает с интерфейсом Windows Forms, который очень удобен в использовании, а также позволяет без проблем создавать приложения.
* JIT-компиляция (Just-In-Time), перекомпиляция исходных файлов во время работы программы.
  1. Интерфейс программы

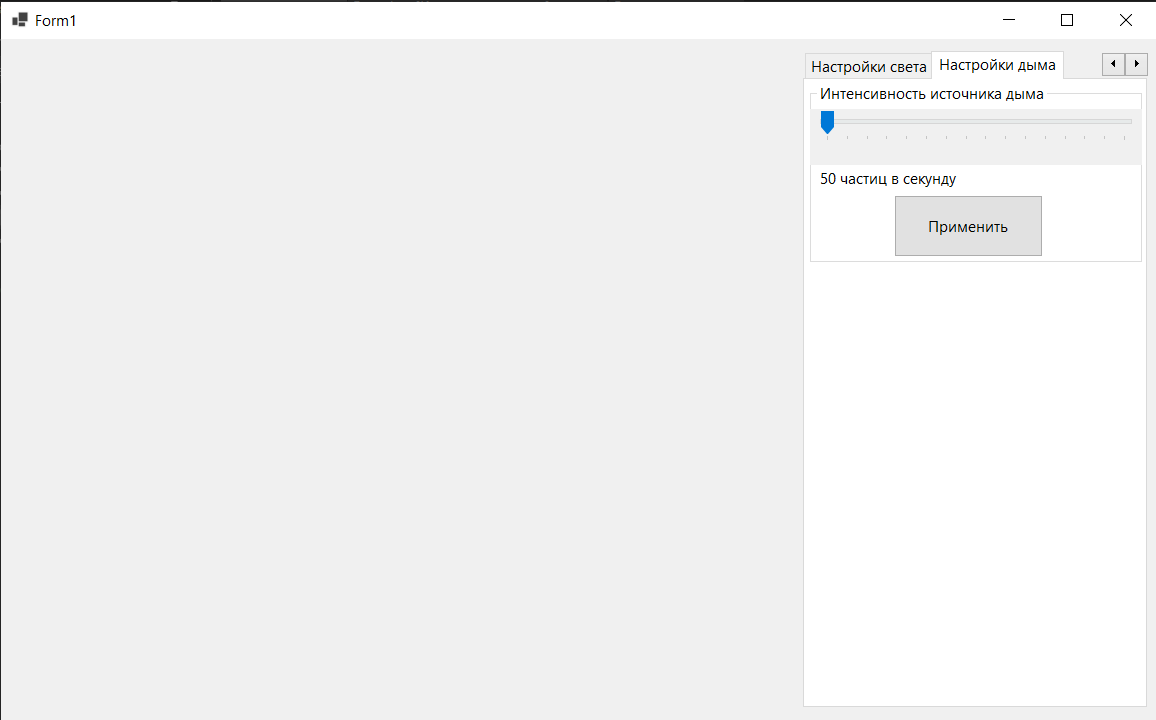
На рисунках 3.1 – 3.4 изображен интерфейс программы

*Рисунок 3.1 – Вкладка настроек сцены*

*Рисунок 3.2 – Вкладка настроек положения объектов на сцене*



*Рисунок 3.3 – Вкладка настроек источника света*

 *Рисунок 3.4 – Вкладка настроек источника дыма*

Заключение

Во время этой учебной практики я смог рассмотреть теоретические вопросы, связанные с разработкой сцены, сравнить и выбрать необходимые для программы алгоритмы, и определиться с выбором языка и среды разработки.