

Министерство образования Республики Беларусь
Учреждение образования
Белорусский государственный университет
информатики и радиоэлектроники

УДК _____

Стаховский
Антон Владимирович

Динамическая симуляция огня

ДИССЕРТАЦИЯ
на соискание степени магистра информатики и вычислительной техники
по специальности 1–40 81 02 Технологии виртуализации и облачных
вычислений

Научный руководитель
Кукин Дмитрий Петрович
кандидат технических наук, доцент

Минск 2020

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ	5
ГЛАВА1 ОБЗОР ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ ПО ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ	6
1.1 Эволюция алгоритмов симуляции огня	6
1.1.1 Истоки. Система частиц	6
1.1.2 Оффлайн симуляция	7
1.1.3 Онлайн симуляция	10
1.2 Обзор и классификация алгоритмов симуляции огня	18
Выводы по главе 1	20
ГЛАВА2 ТЕОРИЯ ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ	21
2.1 Теория компьютерной графики	21
2.1.1 Введение в компьютерную графику	21
2.1.2 Библиотека OpenGL	22
2.2 Физика огня	26
2.2.1 Компоненты горения	26
2.2.2 Воспламенение объектов	27
2.2.3 Классификация огня	28
2.2.4 Визуальное восприятие огня	29
2.3 Особенности рендеринга в реальном времени	31
2.3.1 Моделирование и визуализация огня	32
2.3.2 Анимация огня	35
Выводы по главе 2	35
ГЛАВА3 РАЗРАБОТКА СИСТЕМЫ ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ	36
3.1 Моделирование огня	37
3.2 Анимация пламени	44
3.3 Результаты экспериментов	46
Выводы по главе 3	47
ЗАКЛЮЧЕНИЕ	48
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	49
Список публикаций соискателя	49
Список использованных источников	50

ВВЕДЕНИЕ

Дым и огонь могут значительно влиять на визуальное восприятие объектов сцены, а также влиять на свойства других объектов сцены. По этой причине огонь и дым являются важными составляющими во многих прикладных областях, таких как симуляция полетов, ландшафтный дизайн, анимация и кинопроизводство. В настоящее время крайне популярен жанр фэнтези, в котором зачастую фигурируют множество огненных и огнедышащих существ, таких как драконы, ифриты, птицы феникс. Для создания данных существ в видеоиграх и кино, требуется моделировать поведение огня, которому не существует аналога в реальном мире. Из-за широкого спектра художественных требований, иногда требуется, чтобы огонь вел себя как горящий газ, в других ситуациях нужно, чтобы он вел себя как горящая жидкость либо имел и вовсе фантастический вид. Часто требуется, чтобы огонь взаимодействовал с окружающими его твердыми телами, а также водой и ветром.

Художники и дизайнеры могут составить крайне детализированный визуальный концепт огня и описание его поведения, однако зачастую очень сложно преобразовать данные идеи в соответствующие математические модели для симуляции огня в графических сценах. Анимация и визуализация данного явления сложной задачей и представляет определенный научный интерес.

Первая широко известная работа по симуляции огня была опубликована в 1982 году, и полученная модель была использована в фильме "Звездный путь 2: Гнев Хана" для создания сцены взрыва генезис-бомбы. С развитием вычислительной техники и активные научные исследования позволили значительно увеличить визуальное качество эффектов в кино. Современное состояние работ в данной области можно увидеть в фильме "Хоббит: Битва пяти воинств" в сцене, где Смауг сжигает пламенем Эсгарот. При создании эффектов для кинофильмов главную роль играет визуальное качество эффектов, а также гибкость и простота в настройке параметров, для моделирования огня с необходимым визуальным стилем и поведением. Скорость симуляции является вторичной, создание кадра может занимать несколько часов. Современные алгоритмы, используемые в кино позволяют создавать высокореалистичный огонь, однако применение данных алгоритмов для симуляции в режиме реального времени затруднительно. Например, в видеоиграх необходимо успевать рассчитывать и отрисовывать 60 и более кадров каждую секунду. Малое время на отрисовку кадра и необходимость взаимодействия огня с объектами окружающей среды приводят накладывают дополнительные ограничения на выбор алгоритмов симуляции.

Симуляция трехмерного огня в режиме реального времени находит свое применение в различных интерактивных приложениях. Среди интерактивных приложений, анимации огня наиболее востребованы в видеоиграх. В видео-

играх необходимость симуляции огня была с самого момента их появления, однако, всего два десятилетия назад стало возможным использовать огонь в трехмерных сценах. В компьютерной графике довольно часто требуется найти компромисс между скоростью и реализмом. В приложениях реального времени, скорости отрисовки отдается наибольший приоритет; увеличенный реализм бесполезен, если частота кадров не дотягивает до определенного уровня. Поэтому основной проблемой рендеринга в реальном времени является поиск таких алгоритмов, которые позволяют получить достаточную реалистичность, при которой частота кадров будет не менее минимального порога.

Для каждого из желаемых атрибутов необходимо выбрать алгоритм, который оптимальен для наших целей. Такая оптимальность зачастую приводит к созданию различных ухищрений для достижения желаемого эффекта. Основная идея в том, что если результаты визуально приемлемые, и симуляция идет достаточно быстро, практически никто не заметит особенной разницы. В этом нет никакого стыда, поскольку такие ухищрения являются частью индустрии компьютерной графики с самого ее появления.

За последние 10 лет, значительно возросла производительность компьютеров, особенно стремительно развивались графические карты. Современное аппаратное обеспечение компьютеров, сделало возможным применение более сложных моделей, использование алгоритмов анимации и рендеринга, использование которых раньше было невозможно из-за аппаратных ограничений. В частности это активизировало интерес рендерингу с помощью воксельной графики, некоторые из алгоритмов воксельной графики были разработаны более 30 лет назад, однако, их использование было невозможно либо крайне ограниченно из-за требований к оперативной памяти. В актуальных публикациях используются уравнения Навье-Стокса для вычисления скорости, плотности и температуры частиц огня. Данный метод позволяет добиться высокой реалистичности анимации, однако имеет высокую вычислительную сложность по количеству операций.

В настоящее время разработчики игровых движков исследуют возможности совместного использования новых подходов, таких как воксельная графика, с уже устоявшимися на основе систем частиц. Задача оптимизации и улучшения классических алгоритмов также остается актуальной.

Актуальные алгоритмы анимации огня и современные алгоритмы трехмерного рендеринга являются предметом данной диссертации, цель которой – создание системы для динамической симуляции огня.

Компоненты данной системы в дальнейшем могут быть интегрированы в игровые движки, трехмерные редакторы.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Объект и предмет исследования

Объектом исследования является огонь в трехмерной графике как один из элементов трехмерной сцены.

Предметом исследования является динамическая симуляция объемного огня в режиме реального времени.

Цель и задачи исследования

Цель исследования — разработка системы динамической симуляции трехмерного огня.

Задачи исследования:

1. Обзор и анализ научных работ по современным алгоритмам анимации огня и трехмерному рендерингу.
2. Анализ теории динамической симуляции огня.
3. Реализация системы.
4. Анализ полученных результатов.

Связь с реальным сектором экономики

На основе разработанной системы можно сформировать модули для популярных игровых движков и графических редакторов, что обеспечит доступность системы для широкого круга лиц и позволит распространять систему на соответствующих коммерческих площадках. Полученные модули можно будет легко интегрировать в коммерческие продукты.

Апробация диссертации

Результаты исследований по теме диссертации были представлены в виде доклада "Современные алгоритмы моделирования аморфных объектов" и представлены на 55-ой юбилейной научной конференции аспирантов, магистрантов и студентов БГУИР в 2019 году. Также работа была представлена на 56-ой научной конференции аспирантов, магистрантов и студентов БГУИР в 2020 году в докладе "Динамическая симуляция объемного огня".

Публикация результатов исследований

Результаты исследований были опубликованы в виде тезисов доклада на 55-ой юбилейной научной конференции аспирантов, магистрантов и студентов БГУИР [1–А] и 56-ой научной конференции аспирантов, магистрантов и студентов БГУИР [2–А]. На основе полученных в ходе исследований результатов была опубликована статья "Анализ современных алгоритмов симуляции огня" [3–А].

ГЛАВА 1

ОБЗОР ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ ПО ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ

Моделирование бесформенных объектов, таких как дым, огонь, туман, дымка является предметом постоянных исследований в области компьютерной графики, поскольку данные объекты не имеют четко очерченных границ и являются мобильными по своей природе. Высокая коммерческая ценность данных эффектов в кинематографе и видеоиграх является двигателем постоянных исследований в данной области. Наибольший вызов представляет моделирование данных объектов в реальном времени, где необходимо получить максимально реалистичную симуляцию за время обновления экранного кадра (60Hz+) [1].

1.1. Эволюция алгоритмов симуляции огня

1.1.1. Истоки. Система частиц

Пионером компьютерной симуляции огня является Уильям Томас Ривз. В 1983 году в своей работе он ввел понятие "система частиц" в качестве примитива для моделирования, анимации и рендеринга [2]. В фильме "Звездный путь 2: Гнев Хана" он смоделировал так называемую "расширяющуюся стену огня", созданную с помощью двухуровневой системы частиц. (рис. 1.1 и рис. 1.2).

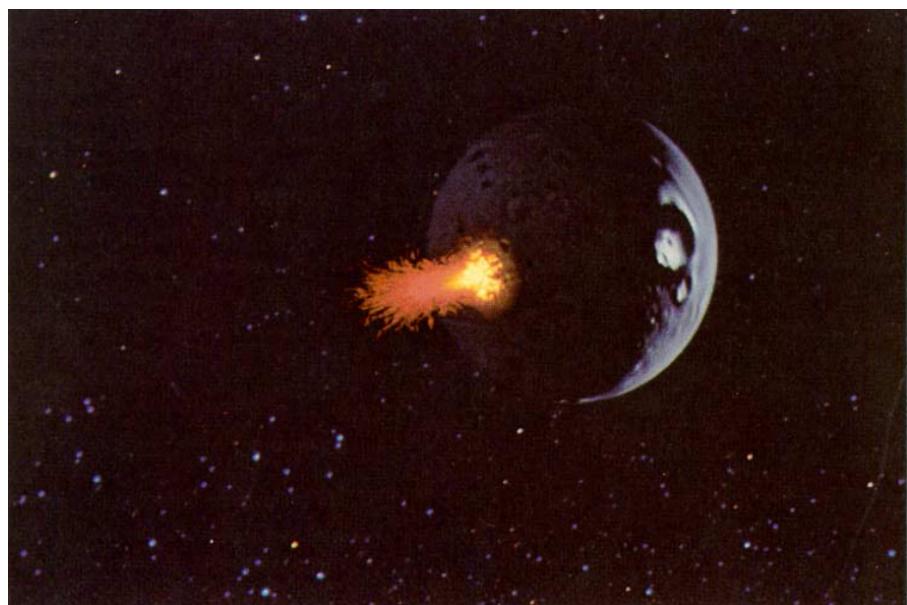


Рисунок 1.1 – Первоначальный взрыв

Система частиц верхнего уровня находилась в центре взрыва генезис-бомбы, она генерировала частицы, которые в свою очередь являлись системами

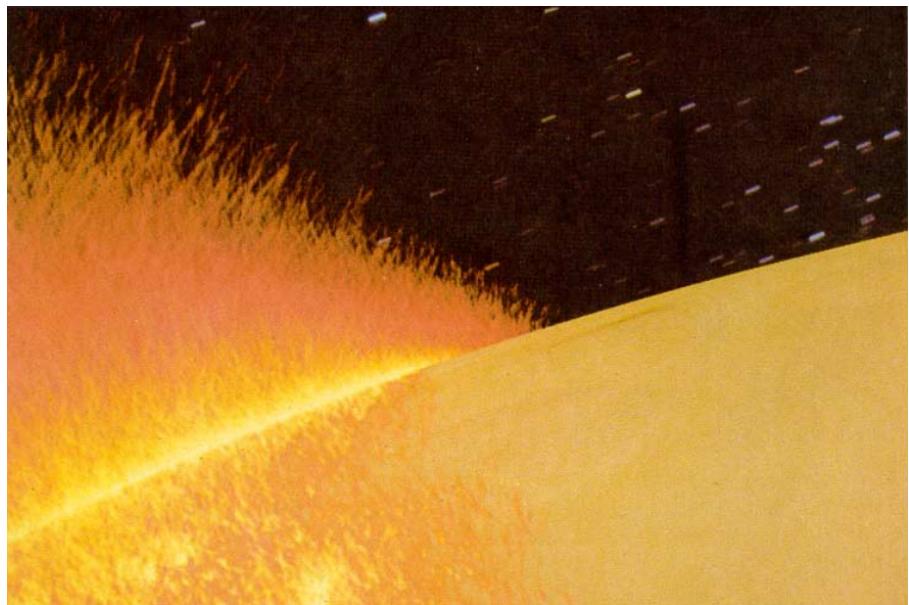


Рисунок 1.2 – Стена огня вот-вот поглотит камеру

частиц. Эти системы частиц использовались для моделирования взрывов, при которых каждая такая система частиц вела себя как небольшой вулкан, извергающийся в сторону распространения взрывной волны и затухающий под воздействием силы гравитации. Поскольку частицы имеют дискретную природу, для достижения хороших результатов потребовалось колоссальное количество частиц. Но, поскольку моделирование в реальном времени не требовалось, это не оказалось проблемой.

1.1.2. Оффлайн симуляция

В данное время наибольшего успеха исследователи добились в нечувствительном к времени симуляции кинематографе. Несмотря на то, что данная работа нацелена на компьютерную графику реального времени, необходимо упомянуть несколько работ в области оффлайн симуляции, поскольку понимание основных идей, заложенных в них, позволило перенести некоторые из них в область графики реального времени.

В публикации 2002 года Дюк Нгуен и его коллеги представили метод моделирования огня, полностью основанный на физико-математическом подходе [3]. В симуляции использовались несжимаемые уравнения Навье-Стокса для горячих газов, это позволило также смоделировать эффект расширения, вызванный испарением, и эффект текучести поднимающихся дыма и сажи. Демонстрация работы системы представлена на рисунке 1.3.

Как видно на рисунке, данная симуляция отличается реалистичным позиционированием и движением газообразных субстанций. Однако данный подход сложно реализовать в рендеринге реального времени, поскольку необходимо находить решение большого количества комплексных уравнений за время



Рисунок 1.3 – Два горящих полена находятся на земле и являются источником топлива. Бревно, лежащее поперек, еще не загорелось, поэтому пламя его обтекает кадра.

Другой выдающейся работой в этой области является статья Арно Ламорлэтта [4], опубликованная в 2002 году. В данной работе автор представил совершенно другой подход в области оффлайн симуляции огня. Данный подход был использован в мультфильме Шрек студии Dreamworks (рис. 1.4). В



Рисунок 1.4 – Кадр из мультфильма Шрек

симуляции Ламорлэтта фокус с реалистичности смешен в сторону на эстеизизма, управляемости и гибкости. Автор руководствовался идеей, что имея задачу предоставить инструмент для художественного и поведенческого контроля, будет крайне сложно достичь поставленной цели полагаясь строго на физические уравнения.

В 2009 году Кристофер Хорват и Вилли Гейгер представили инновационную комбинацию симуляции с помощью крупной решетки частиц и тонко

настроенных визуально-ориентированных улучшений симуляции, рассчитываются на графическом процессоре (ГП) [5]. Полученные изображения имеют поразительную детализацию и могут быть легко интегрированы в кинематографические фотоснимки (рис. 1.5). Данная техника улучшения симуляции ис-



Рисунок 1.5 – Три различных кадра симуляции огня. Быстро движущийся огненный шар с искрами. Извивающийся костер. Плотная стена дыма и огня.

пользует особенности и ограничения зрительного восприятия, а также особенности концентрации внимания зрителя. Множество независимых ГП используются для быстрого увеличения качества изображения, что позволяет достичь очень высокого разрешения.

В статье [6] автор описывает метод анимации приостановленных взрывов. Данный метод использует несжимаемую модель жидкостей для расчета движения воздуха и горючих газов. Горение смоделировано с помощью частиц и систем жидкостей. Полученная система позволяет генерировать кадр взрыва всего за несколько секунд. Сравнение кадра, созданного системой, с реальным взрывом приведено на рисунке 1.6. Принципы, заложенные в данной публика-



Рисунок 1.6 – Сравнение взрыва из симулятора с фотографией взрыва в угольной шахте
ции в дальнейшем были доработаны и реализованы в публикации [7].

Одной из наиболее эффектных сцен с использованием симуляции огня можно назвать сцену сожжения Озёрного города в фильме "Хоббит: Битва пяти воинств" (рис. 1.7). Обзор техник, использованных в фильме приведен в презентации [8]. Взмахи крыльев Смауга приводили в движение симуляцию воздушных потоков. Симуляция воздушных потоков в свою очередь влияла на симуляцию огня, которая оказывала влияние на детализированное разрушение окружения с помощью физики твердого тела. Физика разрушения зданий



Рисунок 1.7 – Сцена сожжения Озерного города

взаимодействовала с симуляцией воздушных потоков, вызывая подъем дыма и пара. Когда дракон выдыхал огонь, происходил расчет всех вышеописанных действий.

1.1.3. Онлайн симуляция

Трехмерный огонь, моделируемый в реальном времени, находит свое применение в интерактивных приложениях. Среди интерактивных приложений можно выделить компьютерные игры, в которых необходимость показывать взрывы появилась практически с самого момента их появления (рис. 1.8).



(а) Огонь создан с помощью заранее нарисованного ядра битовой карты, которое окружают анимированные времени частицы



(б) Объемный факел, созданный из непрозрачных полигонов

Рисунок 1.8 – Скриншоты из Quake (1996)

Компьютерные игры являются основными потребителями графических

компьютерных анимаций огня. Однако это стало возможным лишь пару десятилетий назад. С тех пор скорость аппаратного обеспечения для рендеринга время росла экспоненциально, открывая возможности для все более и более детализированных эффектов. К сожалению, поскольку игры зачастую являются проприетарными по своей природе, литературных источников по алгоритмам, используемых в играх крайне мало [9].

В 2014 году была представлена система NVIDIA FlameWorks [10]. Данная система позволяет добавлять реалистичный огонь, дым и эффекты взрывов в игры. Данная система совмещает передовую симуляцию жидкостей на основе решетки вместе с эффективной системой объемного рендеринга. Все компоненты системы оптимизировано для работы в реальном времени. Все вычисления выполняются на ГП с помощью DirectX 11. На рисунке 1.9 представлен кадры из демонстрационного ролика. В данном ролике использовалась



Рисунок 1.9 – Демонстрация работы NVIDIA FlameWorks

воксельная решетка размерами $512 \times 256 \times 256$ вокселей, каждый кадр обновлялся около 32 миллионов вокселей. Данная симуляция работала с частотой выше 30 кадров в секунду на GeForce Titan.

В работе [11] автор подробно описывает создание симулятора на основе систем частиц. Для структуризации объектов в системе автор использовал четырехуровневую иерархию (частица, эмиттер, семейство эмиттеров, менеджер эмиттеров). Это позволило ему создавать сложное пламя, состоящее из нескольких разнородных источников с различной интенсивностью. Для моделирования автор использовал совместно идеи из термодинамики и идеи из

теории роевого поведения. Использование алгоритма косяка птиц позволило добиться анимации языка пламени как единого целого.

Юрий Ванзин в своей диссертации [12] использовал сплайны для моделирования языков огня. Автор использовал частицы для расчета анимации, которые в дальнейшем использовались в качестве скелета для сплайнов. Для расчета макро-движения огня автор использовал: скорость эмиттера, поля ветров, диффузионное смещение и термальную текучесть. В результате работы, у автора получилось создать полноценный фреймворк для рендеринга огня на основе игрового движка Irrlicht.

В работе [13] работе представлена система, которая предназначена для использования 3D-художниками. Пользователь может задать форму пламени на различных этапах в виде рисунка. Система производит расчет промежуточных кадров и позволяет экспортить результаты для рендеринга в графических редакторах. Данный метод использует в качестве примитива моделирования частицы, рендеринг осуществляется с помощью сплайнов. Данный алгоритм развивает идеи, предложенные ранее в [12]. Для управления анимацией огня авторы используют четыре вида полей ветров (рис. 1.10):

- равномерное;
- вихревое;
- поле-источник;
- поле-сток.

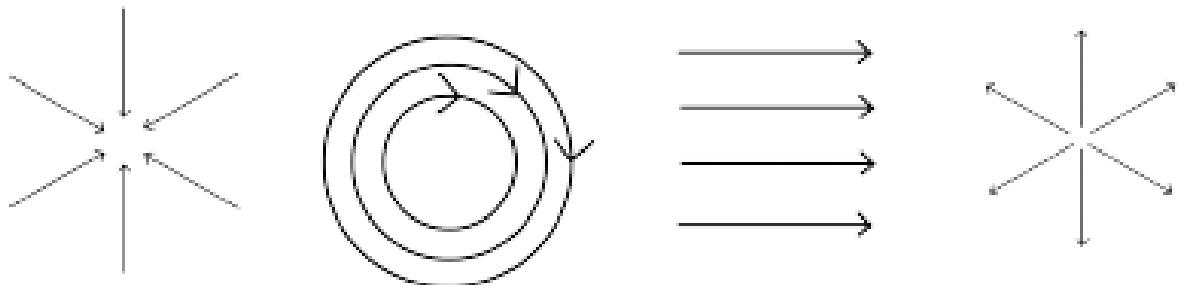


Рисунок 1.10 – Различные виды полей ветров

Интересный подход к проблеме был предложен в [14]. Схема системы приведена на рисунке 1.11. На первом этапе метода происходит сбор инфор-

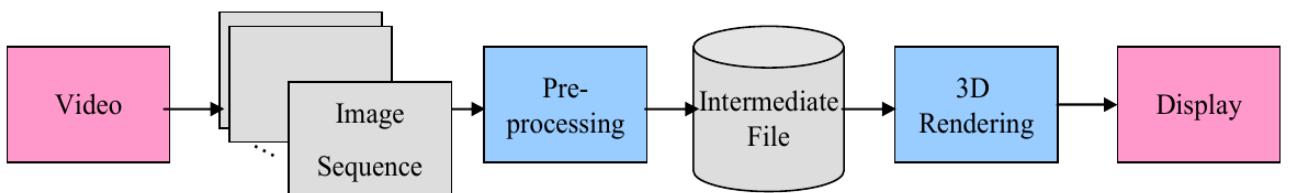


Рисунок 1.11 – Схема системы из [14]

мации из видео с помощью алгоритмов цифровой обработки изображений. Из полученных данных происходит извлечение признаков для обучения модели.

Результат полученный с помощью обученной модели используется совместно с процедурными методами для рендеринга анимации огня. Результаты работы системы выглядят более реалистично по сравнению с традиционными методами, а способность взаимодействовать с окружением находится примерно на уровне других методов реконструкции огня.

В статье [15], описывается симуляция огня с помощью вокселей. Авторами предложены алгоритмы распространения огня и поглощения огнем горючих предметов. Данные алгоритмы предназначены для работы на объемных наборах данных. Языки пламени в предложенном методе создаются на основе метода решеточных уравнений Больцмана (LBM). Полученная авторами система позволяет моделировать распространение огня, горение объектов. Работа системы продемонстрирована на рисунке 1.12. На рисунке изображен объемный



Рисунок 1.12 – Воксельное пламя

стол. Внутренняя основа из негорючего металла открывается по мере того, как сгорает внешний слой из дерева.

Статья [16] описывает методы ускорения симуляции огня с помощью технологий параллельных вычисления CUDA, также авторы исследовали поведение системы с различными примитивами рендеринга (точки, треугольники, текстуры). Полученные результаты показывают в среднем улучшение на 10% при использовании параллельного кода на CUDA. Рендеринг с помощью точек показал наивысшую производительности, однако рендеринг с помощью текстур показал лучшие визуальные результаты. По результатам опытов было найдено оптимальное ядро 128×128 , что соответствует 16384 частицам в системе.

В работе [17] метод анимации огня на многогранных поверхностях. Для расчетов в данном методе используются простая геодезия на мешах, скорость, зависящая от внешнего ветра или сил, угол наклона, и другие атрибуты. В от-

личие от предыдущих методов, данный метод позволяет выполнять отложенный адаптивный расчет фронта огня, метод также обрабатывает точкистыка, объединение фронтов и поддерживает удаленное возгорание. Все эти приемы помогают достаточно реалистично отобразить поведение реального огня.

Авторы работы [18] предлагают в своей статье использовать текстурные сплэты в качестве базового примитива рендеринга. Для реализации взаимодействия огня с ветром и окружающими объектами авторы использовали простую линейную LBM. Результаты симуляции можно увидеть на рисунке 1.13. Полу-



Рисунок 1.13 – Симуляция огня в системе [18]

ченная ими система имеет следующие преимущества:

1. Тurbulentное поведение огня может быть выражено с помощью текстурных сплэтов.
2. Расчеты LBM могут быть выполнены на ГП.

В недавней статье [19] авторы предлагают новый гибридный метод симуляции для моделирования процессов горения твердого топлива. Авторы создали гибридную систему на основе метода решеточных уравнений Больцмана и уравнений Навье-Стокса (решетка Эйлера). В данной работе показано, что вычислитель на основе уравнений Навье-Стокса может быть использован для расчета участков огня, находящихся в воздухе. Для расчета поведения огня внутри и на поверхности твердых тел авторы рекомендуют использовать LBM.

Авторы работы [20] предлагают простой и эффективный подход для создания высокодетализированных 3D анимаций из низкодетализированных симуляций на основе физики жидкостей. Создатели алгоритма используют заранее рассчитанную базу полей скорости, полученных из двумерных симуляций на основе физики жидкостей. Во время работы программы выполняется

низкодетализированная 3D симуляция, поля скоростей симуляции каждый шаг аппроксимируются с помощью линейной комбинации заранее рассчитанных полей. С помощью такого метода пользователи могут разрабатывать низкодетализированные анимации и добавлять к ним детали на завершающих этапах (рис. 1.14).

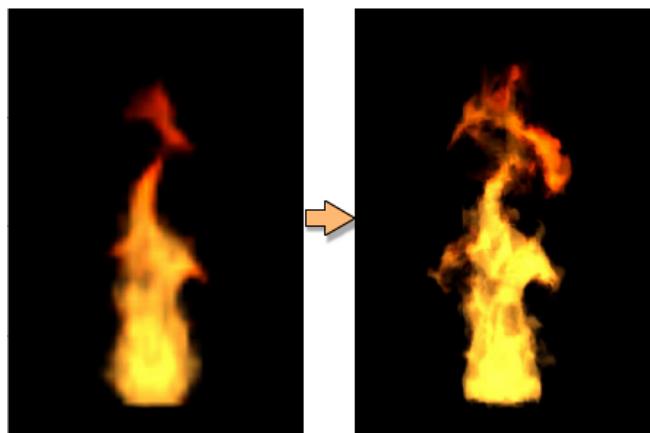


Рисунок 1.14 – Преобразование огня с разрешением $32 \times 32 \times 64$ точек в огонь с разрешением $128 \times 128 \times 256$ точек

В статье [21] авторы представляют один из методов томографической реконструкции огня. Структура метода представлена на рисунке 1.15. С помо-

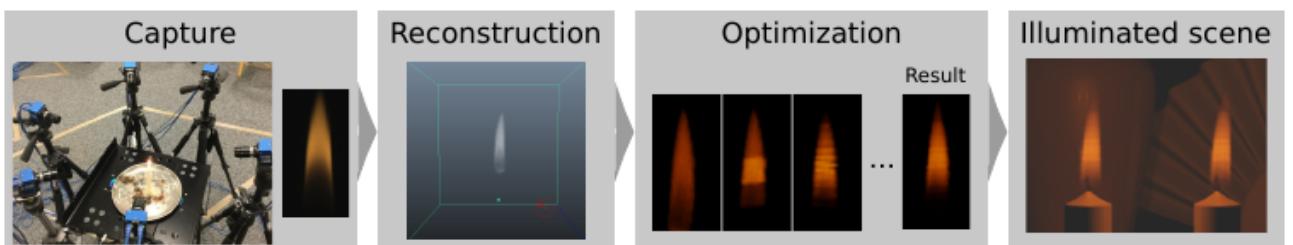


Рисунок 1.15 – Стадии метода реконструкции огня из [21]

щью RGB камер происходит запись анимации реального огня. Далее на стадии реконструкции происходит расчет плотности объема занимаемого огнем. На стадии оптимизации происходит вычисление температур участков пламени и расчет полей плотности горючего. Рассчитанные модели в дальнейшем могут быть использованы для создания реалистичного освещения в виртуальных сценах. Отличием данной работы от предыдущих решений является то что созданный огонь дает реалистичное освещение сцены без необходимости искусственного создания дополнительных источников освещения.

В работе [22] приводится исчерпывающее описание использования диффузионных процессов для симуляции газообразных феноменов. Авторы представили простую модель анимации и распространения огня, также ими была разработана эффективная реализация алгоритма глобального освещения в присутствии газов и огня. В отличие от предыдущих работ в данной области, данная работа предоставляет дополнительный уровень контроля пользователю над

развитием газообразного феномена. Такие параметры, как позиционирование полей ветров, их размер и магнитуда могут быть изменены в режиме реального времени.

Интересная термодинамическая модель пламени предложена в работе [23]. Авторам удалось создать быструю интерактивную модель симуляции огня, предоставляющую высокий уровень реалистичности. В данной системе процесс горения симулируется с помощью поглощения топлива и окисляющих газов в каждой ячейке сетки сцены. Процесс горения вызывает выделение тепла, распределение и распространение которого моделируется в системе. Процесс распространения тепла создает конвекционные потоки в воздухе, придавая соответствующую форму пламени. Кроме моделирования распространения тепла, в данной системе также симулируется процесс распространения огня и процессы самовозгорания твердотопливного горючего.

Использование симуляции огня для создания стилизованных арт-работ приведено в диссертации [24]. Автору удалось создать простую для использования художниками систему. Пользователи могут создавать и анимировать контрольные линии в привычных им пакетах и импортировать результаты в систему. Симулятор использует вычислитель для уравнений стационарных жидкостей и показывает довольно быструю работу на относительно низких разрешениях (100 вокселей на слайд), что позволяет выполнять визуализацию в реальном времени с помощью OpenGL . Результаты работы системы представлены на рисунке 1.16.

В статье [25] представлена простая и быстрая реализация вычислителя динамики жидкостей для игровых движков. Данный инструмент основан на уравнениях Навье-Стокса для расчета поведения жидкостей. Вычислитель, предложенный в работе в первую очередь нацелен на визуальное качество. Авторы демонстрируют простоту их реализации, предоставляя в публикации полный код симулятора на языке Си. Предложенный алгоритм может использоваться для расчета сеток разумного размера в реальном времени, используя для этого обычный персональный компьютер.

В работе [26] приведено описание эффективных методов для создания реалистичных изображений природных феноменов. В данной работе приведены алгоритмы и уравнения для создания дыма, облаков, неба. В качестве примитива симуляции выбраны частицы. Описанные в статье методы могут быть использованы для получения фотoreалистичных изображений.

В презентации [27] приводится хороший обзор истории симуляции огня и состояния проблемы на сегодняшний день. Авторы приводят код простого вычислителя давления жидкостей на CUDA. Также авторы дают ряд подсказок по симуляции огня, позволяющих добиться максимальной реалистичности симуляции. Среди этих советов можно особенно выделить следующие:

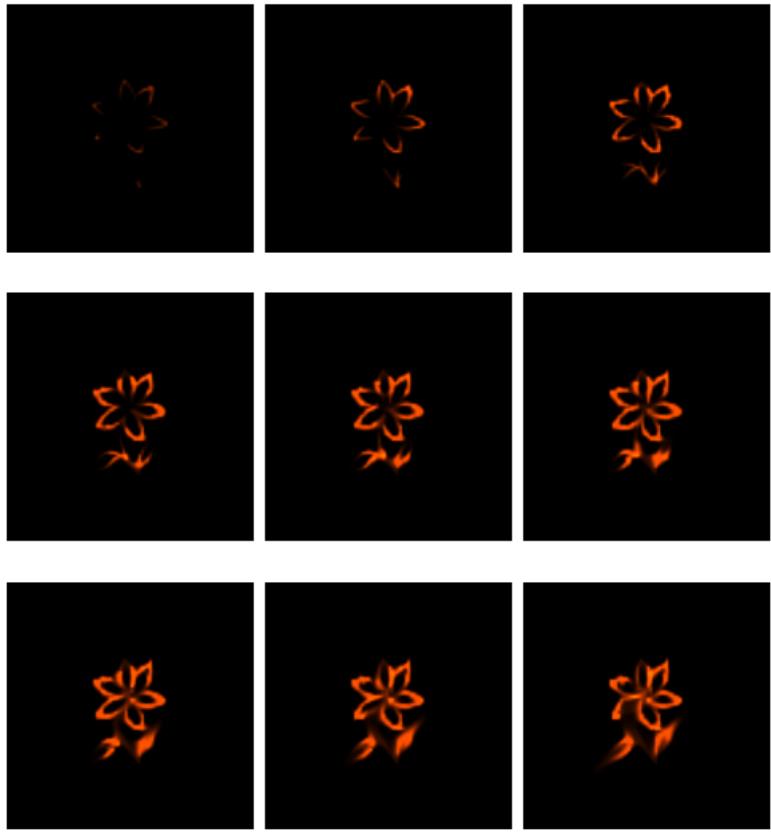


Рисунок 1.16 – Рендеринг огненного цветка

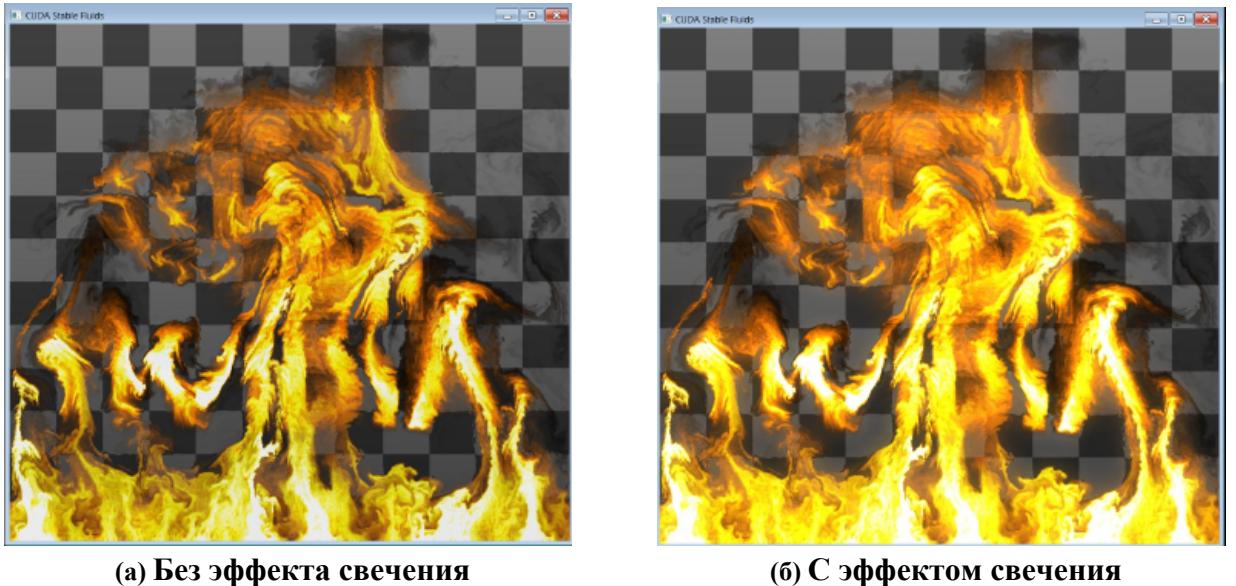
1. Постобработка крайне важна. С помощью таких приемов как размытие в движении и свечение объектов (рис. 1.17) можно лучше доносить зрителю информацию о температуре.

2. Добавление шумов. Добавление процедурного шума позволяет создать эффект турбулентности и увеличить реалистичность.

3. Искры. Для добавления искр авторы советуют использовать четырехугольники, растянутые от прошлой позиции до текущей. Для этого рекомендуется использовать геометрический шейдер.

Авторы статьи [28] представили в 2007 году метод для генерации процедурного объемного огня в режиме реального времени (рис. 1.18). С помощью комбинации кривообразной объемной свободной деформации, рендеринга с использованием аппаратного ускорения и М-шума авторам удалось создать подрагивающий огонь с уникальной анимацией. Данный метод поддерживает эффективное обнаружение коллизий, что, вкупе с хорошо спроектированной системой частиц, позволяет добиться двухстороннего интерактивного взаимодействия между огнем и окружающей его средой.

В статье [29] представлен метод симуляции трехмерного огня в режиме реального времени (рис. 1.19). Идею данного метода авторы взяли из старого механического приема под названием "шелковый факел". Авторы выполняли моделирование кинематики огня с помощью уравнений гармонических колебаний, а турбулентность и визуальную динамику с помощью последовательно-



(а) Без эффекта свечения

(б) С эффектом свечения

Рисунок 1.17 – Влияния эффекта свечения на симуляцию огня

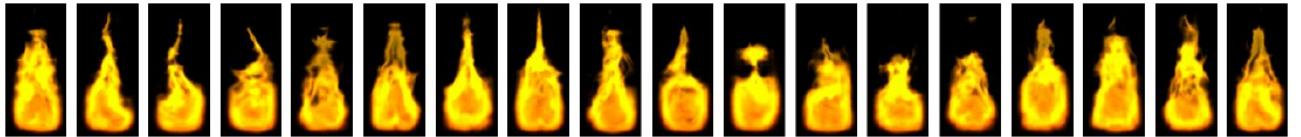


Рисунок 1.18 – Процедурный шум используется для придания каждому огню уникального вида

стей текстур с различной прозрачностью. Текстуры определялись скоростью топливных газов.

1.2. Обзор и классификация алгоритмов симуляции огня

В предыдущих разделах было описано множество различных методов симуляции огня, однако, можно обобщить данные методы и выделить отдельные классы. Различные методы применяемые при симуляции огня можно разделить на следующие группы:

- текстурный маппинг;
- системы частиц;
- физико-математические методы;
- клеточные автоматы;
- томографическая реконструкция и др.

В работе [30] представлен детальный обзор различных техник для создания реалистичной симуляции огня. В данной работе проведен сравнительный анализ нескольких техник симуляции огня, включая симуляцию с помощью вокселей и симуляцию с помощью стационарных жидкостей. Приведено краткое описание каждой техники, с описанием их преимуществ и недостатков.

В 2011 году Чжао Хуэй и его коллеги представили статью [31], авторы проанализировали большое количество методов симуляции огня и смогли вы-

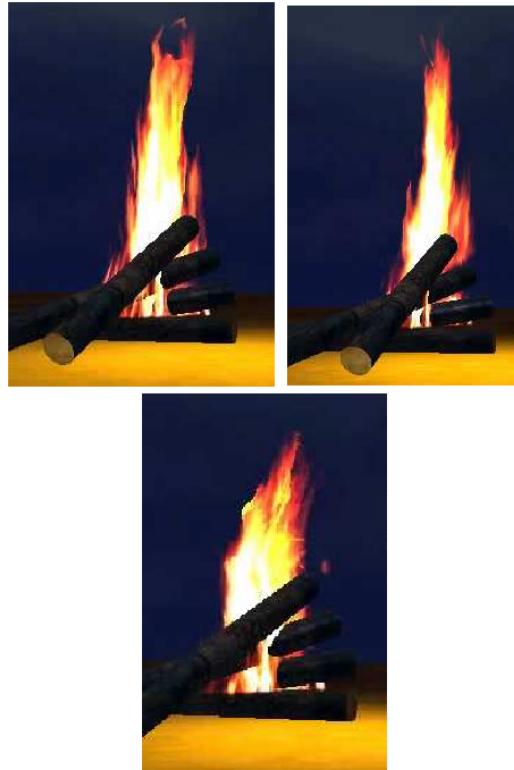


Рисунок 1.19 – Костер, созданный с помощью 8 различных текстур

двинуть свою классификацию методов. Авторами был проведен анализ наиболее популярных методов по следующим критериям:

- применимость в реальном времени;
- степень реалистичности;
- пространственно-временная сложность;
- настраиваемость;
- интерактивность.

Результаты данного исследования можно увидеть в таблице 1.1.

Таблица 1.1 – Сравнение производительности различных методов симуляции огня

	Real-time	Realistic	Spatio-temporal complexity	Editability	Interactivity
Texture mapping	High	Low	Low	Low	No
Particle system	Inversely with particles count	Medium	Proportional to particles count	Random large and difficult to control	Medium
Mathematical physics-based	Low	High(physical consistency)	High	Parameter control	High
Cellular automation	Inversely proportional to the complexity of combined requirements	Have certain realistic	Cell simple but the combined complex	Modium	Limited
Tomographic reconstruction	No	High(visual consistency)	Data acquisition and processing complex	No	No

В процессе исследования работ по симуляции огня, авторы смогли выделить следующие проблемы:

1. Системы частиц демонстрируют определенный уровень детализации огненной сцены, особенно удачно они отражают случайные изменения в огне.

Однако, их поведение слишком непредсказуемо, чтобы достичь точного отображения огня.

2. Математические методы, основанные на физике, предоставляют большие преимущества в реалистичности, настраиваемости и интерактивности, но они с трудом подходят к применению в реальном времени.

3. Метод клеточных автоматов лучше всего подходит для моделирования распространения огня, поскольку пространственно-временная сложность данного метода сильно ограничена из-за необходимости комбинации клеток.

4. Методы томографической реконструкции по изображению позволяют генерировать точные анимации огня, которые могут быть отрисованы в произвольных позициях. Данные методы не могут предоставить широкие возможности по настраиваемости и интерактивности. Вдобавок к этому, цены на оборудование для сбора данных крайне высоки.

Выводы по главе 1:

1. При выборе алгоритма для динамической симуляции необходимо найти баланс между реалистичностью симуляции и скоростью симуляции. Частота кадров не должна падать ниже заданного минимального порога.

2. Большую роль в создании реалистичной симуляции играет постобработка. Такие вещи, как добавление шумов и размытия позволяют существенно увеличить реализм сцены.

3. Моделирование огня с помощью системы частиц до сих пор является наиболее популярным методом, который позволяет обеспечить средний уровень реалистичности, но при небольшом количестве частиц обеспечивает высокую скорость симуляции. Данный метод будет использован для моделирования частиц в рамках диссертации.

4. Использование алгоритмов из области физики, в особенности термодинамики позволяет создавать реалистичные симуляции. Некоторые из физических алгоритмов хорошо показали себя на практике и будут использованы для создания симулятора в рамках диссертации.

ГЛАВА 2

ТЕОРИЯ ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ

В данной главе будут приведены основные теоретические сведения, которые необходимы для введения в практическую часть диссертации. В главе будут приведены необходимые определения из областей компьютерной графики, физики математической симуляции, будут представлены рисунки и уравнения, поясняющие информацию.

Глава состоит из трех разделов:

1. В разделе "Теория компьютерной графики" приводятся определения терминов из области компьютерной графики, которые будут использоваться далее в тексте диссертации. Также в разделе приведен обзор преимуществ и особенностей библиотеки OpenGL, которая будет использована для создания практической части диссертации.

2. Раздел "Физика огня" описывает огонь как физический процесс. В разделе будет дано описание процесса горения, приведено отличие между терминами "огонь" и "пламя", дано объяснение некоторым особенностям внешнего вида и поведения огня.

3. В разделе "Динамическая симуляция огня" можно найти описание структуры симуляции огня и ее элементов. Каждая из задач симуляции будет подробно описана. В разделе присутствует описание преимуществ и недостатков популярных алгоритмов в области динамической симуляции огня.

2.1. Теория компьютерной графики

2.1.1. Введение в компьютерную графику

Для начала необходимо дать определение компьютерной (или машинной) графике.

Компьютерная графика — это совокупность технических, математических и программных средств и приемов, позволяющих осуществлять ввод и вывод из ЭВМ графической информации без ручного преобразования информации в числовую или графическую форму [32].

Можно выделить три основных этапа формирования изображения электронной вычислительной машиной:

- построение модели объекта или сцены, содержащей несколько объектов (т.е. описание объектов и их связей в рамках евклидовой геометрии);
- подготовка модели к визуализации в зависимости от местонахождения

наблюдателя (выполнение геометрических преобразований, удаление невидимых линий);

– визуализация с помощью заданного устройства отображения (отсечение по объему/окну видимости, формирование растрового представления, наложение текстуры, затенение, добавление дополнительных эффектов, вывод на терминал).

Изображение можно представить в виде множества точек, линий, строк текста и закрашенных областей, называемых **примитивами**. При этом изображение чаще всего описывается набором вершин, ребер и граней, формирующих в итоге множество прямоугольников с заданными атрибутами и представляющих в совокупности один или несколько объектов сцены. Следует отметить, что визуализация изображений, как правило, выполняется на плоскости, т.е. итоговое изображение двухмерно, и, таким образом, при отображении трехмерных объектов одним из обязательных шагов является проективное преобразование.

Периферийные устройства вывода делятся по своему типу на растровые и векторные. В настоящее время визуализация изображений производится в большинстве случаев именно растровыми устройствами вывода. В растровых устройствах отображения точку заменяет пиксель (от английского *picture element*). **Пиксель** — это наименьшая часть изображения, с которой может работать алгоритм обработки либо визуализации изображения.

Изображения, отображаемые на растровых устройствах, либо хранимые в виде двухмерного массива значений пикселей — раstra, называются **растровыми**. Процесс преобразования векторных моделей в растровые изображения называется **растеризацией**.

2.1.2. Библиотека OpenGL

В практической части диссертации для создания трехмерной (3D) графики используется библиотека OpenGL. **OpenGL** — это программный интерфейс, который позволяет приложениям пользоваться и управлять графической подсистемой устройства, на котором работает OpenGL [33].

OpenGL может работать на различных устройствах от дорогостоящих профессиональных рабочих станций до обычных настольных компьютеров, от игровых консолей до мобильных телефонов. OpenGL предлагает стандартизованный интерфейс (API), который предоставляет широкую портируемость и позволяет разработчикам приложений фокусировать свое внимание на создании качественных продуктов, разработке интересного контента, и увеличении производительности своих приложений вместо того, чтобы беспокоиться о спецификациях платформы, для которой они создают приложение.

OpenGL может разбивать поток работы на фундаментальные элементы

и выполнять их параллельно. Комбинации конвейеризации и параллелизма позволяет получить максимальную производительность от современных графических процессоров. Структура графического конвейера представлена на рисунке 2.1. На рисунке 2.1 блоки со скругленными краями представляют со-

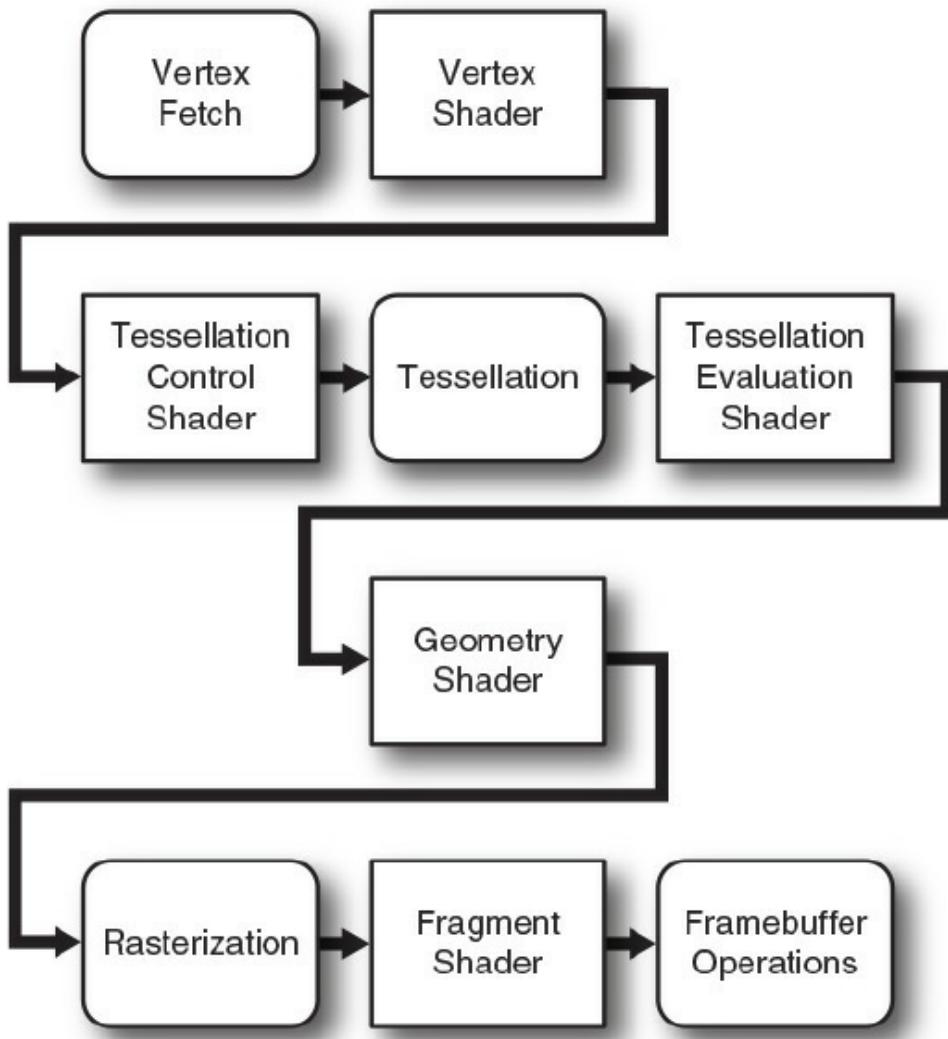


Рисунок 2.1 – Упрощенная структура графического конвейера

бой фиксированные части, которые обычно реализованы как часть драйвера, прошивки или другого системного ПО. Блоки с прямоугольными краями являются программируемыми, это означает, что они могут выполнять шейдеры, предоставляемые разработчиками ПО.

На момент написания диссертации существует 20 изданий спецификации OpenGL [34]. Номера версий и даты их публикации приведены в таблице 2.1. Старые версии OpenGL предлагали непосредственный режим работы (фиксированный конвейер), который был простым способом для отрисовки графики. Однако, большинство функционала было спрятано в библиотеке и у разработчиков не было большого контроля над вычислениями. Начиная с версии 3.2 спецификации началось стимулирование перехода разработчиков к новому API (core профиль), который является версией спецификации, в кото-

Таблица 2.1 – Версии OpenGL

Версия	Дата публикации
OpenGL 1.0	Январь 1992
OpenGL 1.1	Январь 1997
OpenGL 1.2	Март 1998
OpenGL 1.2.1	Октябрь 1998
OpenGL 1.3	Август 2001
OpenGL 1.4	Июль 2002
OpenGL 1.5	Июль 2003
OpenGL 2.0	Сентябрь 2004
OpenGL 2.1	Июль 2006
OpenGL 3.0	Август 2008
OpenGL 3.1	Март 2009
OpenGL 3.2	Август 2009
OpenGL 3.3	Март 2010
OpenGL 4.0	Март 2010
OpenGL 4.1	Июль 2010
OpenGL 4.2	Август 2011
OpenGL 4.3	Август 2012
OpenGL 4.4	Июль 2013
OpenGL 4.5	Август 2014
OpenGL 4.6	Октябрь 2017

рой убрана вся устаревшая функциональность. Начиная с версии 3.3 радикальных изменений в спецификации не происходило, в последующих версиях были добавлены некоторые функции для более удобного выполнения частых задач. Разработанная в ходе диссертации система использует спецификацию OpenGL версии 4.5, однако может быть легко портирована и на более ранние версии.

Необходимо вернуться к обзору графического конвейера и дать определение понятию "шейдер". **Шейдеры** — это небольшие программы, которые предназначены для выполнения на графическом процессоре [35]. Шейдеры могут одновременно выполняться на тысячах вычислительных ядер ГП. Для вывода изображения на экран необходимо наличие в шейдерной программе как минимум вершинного и фрагментного шейдеров. Для написания шейдеров на OpenGL используется **OpenGL Shading Language (GLSL)**. GLSL — Си-подобный язык, который также является частью спецификации OpenGL.

Одну из особенностей OpenGL является использование пяти различных координатных систем:

- локальное пространство (или пространство объекта);
- мировое пространство;
- пространство наблюдателя;
- пространство отсечения;
- экранное пространство.

Для преобразования координат из одного пространства в другое, исполь-

зуется несколько различных матриц трансформации, среди которых, самыми важными являются матрицы модели, вида и проекции. Координаты вершин появляются в локальном пространстве как локальные координаты, и в дальнейшем преобразуются в мировые координаты, потом в координаты вида, отсечения, и, наконец, все все координаты приводятся к экранному пространству. На рисунке 2.2 представлена вся последовательность преобразований и эффект, который оказывает каждое преобразование:

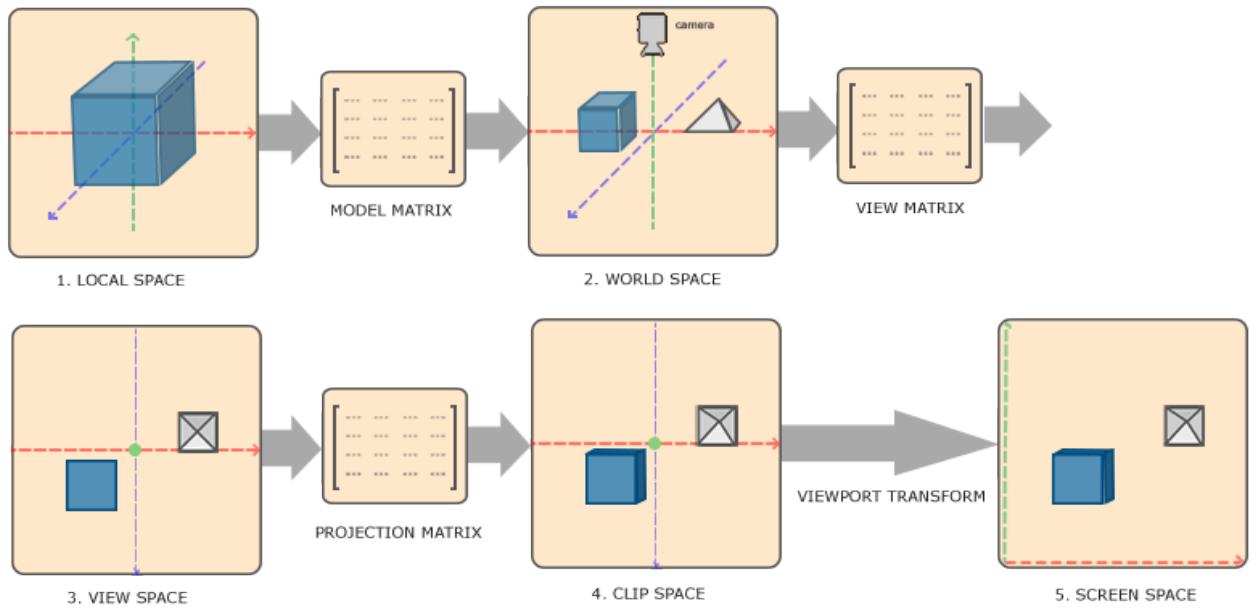


Рисунок 2.2 – Преобразования систем координат в OpenGL

- Локальные координаты это координаты объекта измеряемые относительно точки отсчета расположенной в центре объекта.
- На следующем шаге локальные координаты преобразуются в координаты мирового пространства, которое содержит все объекты сцены. Эти координаты измеряются относительно глобальной точки отсчета, единой для всех объектов расположенных в мировом пространстве.
- Далее происходит преобразование мировых координат в координаты пространства наблюдателя таким образом, что каждая вершина становится видна как если бы на нее смотрели из камеры или из местоположения наблюдателя.
- После того, как координаты были преобразованы в пространство наблюдателя, необходимо спроектировать их в координаты пространства отсечения. Координаты пространства находятся в диапазоне от -1.0 до 1.0 и определяют, какие вершины появятся на экране.

- И, наконец, происходит преобразование координат в экранное пространство. Размер пространства определяется размером окна, в котором происходит рендеринг изображения. В качестве точки отсчета берется нижний левый угол окна. Полученные координаты отсылаются растеризатору для пре-

вращения их во фрагменты (пиксели).

Операцию приведения координат из пространства в модели в пространство отсечения можно выполнить с помощью формулы (2.1).

$$V_{clip} = M_{proj} \cdot M_{view} \cdot M_{model} \cdot V_{local} \quad (2.1)$$

где V_{clip} — координаты вершины в пространстве отсечения;

M_{proj} — матрица преобразования в пространство отсечения;

M_{view} — матрица преобразования в пространство наблюдателя;

M_{model} — матрица преобразования в мировое пространство;

V_{local} — координаты вершины в локальном пространстве.

Матрицы не обладают свойством коммутативности, поэтому важно выполнять умножение в правильном порядке. Умножение матриц следует читать в обратном порядке, т.к. сначала к вершине применяется самая правая операция.

2.2. Физика огня

Перед тем, как приступить к симуляции огня, нужно глубже разобраться в природе исследуемого объекта. Необходимо выделить разницу между огнем и пламенем, дать определение процессу горения и описать его компоненты, сделать обзор основных видов огня и глубже разобраться в физике процесса горения.

2.2.1. Компоненты горения

В разговорном русском языке нет четкого смыслового разделения между словами "пламя" и "огонь" [36], однако в зарубежной литературе присутствует четкое разделение понятий огонь (fire) и пламя (flame). Также в некоторых технических русскоязычных словарях приводится трактовка этих понятий.

Несмотря на то, что стандарт СТ СЭВ 383–87 [37] уже устарел, в нем дается точное определение для ключевых терминов, используемых в диссертации. В следующих подразделах также будут приведены определения из данного стандарта.

Огонь — процесс горения, сопровождающийся пламенем или свечением.

Пламя — зона горения в газовой фазе с видимым излучением.

Среди процессов химических веществ бывают случаи, когда вещество сгорает без пламени [38].

Обычно люди воспринимают горящий объект, словно объект горит сам по себе. Однако, на самом деле пламя образует не сам объект, а топливо, которое он выделяет в окружающую среду. Топливо подымается над поверхностью

объекта из-за тепла, испаряется, вступает в контакт к кислородом и воспламеняется. Таким образом, огонь нуждается в совместном присутствии трех элементов: топливо, тепло и кислород [39]:

1. *Топливо.* Топливом может быть твердая, жидкую либо газообразная субстанция, также топливо должно химически разлагаться на газы или пар. Процесс разложения происходит под воздействием тепла.

2. *Тепло.* Тепло — это мера молекулярной активности в объекте, увеличение температуры ведет к увеличению скорости движения молекул. Если к объекту приложено достаточно тепла, молекулы начинают двигаться настолько быстро, что могут покидать поверхность объекта. Так происходит процесс перехода топлива в газообразное состояние.

3. *Кислород.* Кислород является окисляющим агентом и поэтому необходим для процесса горения. На определенном уровне, частицы огня могут превращаться в частицы дыма из-за нехватки кислорода в воздухе, что вызывает неполное окисление.

2.2.2. Воспламенение объектов

Горение — экзотермическая реакция окисления вещества, сопровождающаяся по крайней мере одним из трех факторов: пламенем, свечением, выделением дыма [37].

Процесс горения имеет следующие стадии:

1. Окисление вызывает разложение горючего материала, происходит медленное выделение газов, включая водяной пар. Процесс протекает непрерывно все время, в течение которого объект взаимодействует с окисляющим агентом, которым может выступать кислород. При температуре окружающей среды окисление обычно происходит настолько медленно, что даже незаметно человеку. Горючие газы пока еще не могут воспламеняться на этой стадии.

2. Скорость окисления возрастает с ростом температуры, в это время некоторые газы становятся воспламеняемыми. Точка, когда в воздухе находится достаточное количество пара, чтобы создать горючую смесь в воздухе, называется *точкой вспышки*.

3. В вышеупомянутой точке в выделенных газах присутствует слишком много углекислого газа и водяного пара, чтобы поддерживать пламя продолжительное время. Однако, тепло огня служит началом для вторичного процесса разложения, которое приводит к процессу устойчивого горения. Эта точка называется *точкой воспламенения*, и она обычно находится на несколько градусов выше точки вспышки.

4. Окисление может идти настолько быстро, что оно покрывает всю поверхность топлива и блокирует доступ к кислороду, препятствуя горению топлива и затрудняя проникновение тепла. Это задерживает распространение тем-

пературы воспламенения вглубь горючего материала. Однако, с увеличением температуры топливо начинает светиться, воздух поступает внутрь для поддержания горения, и происходит горение топлива совместно с выделяемыми газами.

5. Если тепла выделяется больше, чем теряется из-за проводимости, конвекции или излучения, огонь будет поддерживаться и появится пламя. Горение будет поддерживаться, пока либо тепло, горючее, или окисляющий агент не будут убраны из системы.

2.2.3. Классификация огня

Следуя классификации, предложенной в [40], огонь может быть разделен по визуальным характеристикам на следующие группы:

1. *Спокойный огонь*. Типичным примером спокойного огня является огонь свечи. Спокойный огонь редко нарушается воздействием внутренних либо внешних сил. Таким образом, огонь остается спокойным и практически не движется, если движется вообще.

2. *Свободно питаемый огонь*. Свободно питаемый огонь не испытывает нехватки топлива и получает достаточное количество кислорода, также количество топлива и кислорода варьируется в зависимости от времени и местоположения. Это существенно отличается он спокойного огня, который имеет более или менее постоянное и контролируемое количество топлива и доступного кислорода на протяжении всего времени горения и во всех областях огня.

3. *Сильный огонь*. В сильном огне, внешние силы оказывают сильное влияние на огонь, вызывая неравномерную подпитку огня и распространение топливных компонентов в направлениях, отличных от вертикального. Сильный огонь крайне переменчив и крайне непредсказуем из-за сложности лежащих в его основе систем и большого количества влияющих сил.

4. *Огонь, подаваемый под давлением*. Огонь, подаваемый под давлением — это свободный огонь, вектора скорости которого направлены в сторону, в которую происходит выброс топлива в воздух.

5. *Детонации и взрывы*. *Взрыв* — это процесс, в котором за короткое время в ограниченном объеме выделяется большое количество энергии и образуются газообразные продукты взрыва, способные совершить значительную механическую работу или вызвать разрушения в месте взрыва [41]. В *детонациях* используется изначально сжатое топливо; возгорание вызывает взрывную волну, которая вызывает серию небольших взрывов, которые создают новые взрывные волны.

Также хочется отметить такой процесс, как пожар. **Пожар** — неконтролируемое горение, приводящее к ущербу. При моделировании пожаров основной упор делается на расчет скорости распространения дыма и огня, вместо

реалистичной визуализации зачастую используется схематичная. Моделирование пожаром находится за пределами данного исследования.

2.2.4. Визуальное восприятие огня

Огонь является довольно сложным природным феноменом. В данном подразделе будет приведен обзор основных физических особенностей, влияющих на визуальное восприятие человеком огня. В данном разделе будут приведены моменты касательно спектра огня, сопротивления воздуха, вызывающего эффект подергивания, формирования и вида сажи и дыма.

Обычно в естественных науках огонь рассматривают в качестве излучателя черного тела, для описания цвета. Черное тело испускает фотоны различных цветов, от черного до белого, проходя через красный, оранжевый и желтый. Цветовой спектр излучения черного тела приведен на рисунке 2.1. Данный спектр несколько ограниченно описывает световое излучение, испускаемое в процессе горения, поскольку он не учитывает вклад различных примесей, которые могут сильно влиять на цвет пламени.



Рисунок 2.1 – Спектр излучения черного тела

Взаимодействие цвета и температуры можно рассмотреть на примере горения свечи, представленной на рисунке 2.2. Часть пламени, наиболее близкая



Рисунок 2.2 – Горение свечи

к фитилю, невидима, изменяясь постепенно к голубоватому цвету на границе и у основания пламени. В самой горячей точке пламя приобретает белый цвет. Остальные регионы пламени имеют светло-желтый цвет, который постепенно переходит в оранжевый или красный у вершины. Быстрое сравнение цветов пламени свечи и цветовой палитры излучения черного тела показывает

ограничения палитры, поскольку она не нацелена на отображение пламени на восковом топливе.

Необходимо также описать факторы, влияющие на движение огня. Следующий пример, иллюстрирующий движение горячего турбулентного газа был взят из [42]. Представим старомодный паровой двигатель, выпускающий струю горячего пара из своего бойлера. В самом начале, ведущим фактором, влияющим на движение газа, является скорость, с которой его выпустили в окружающую среду. В то время, как пар смешивается с более медленно движущимся воздухом, пар испытывает *сопротивление воздуха* (drag), и начинает вращаться в некоторых местах. Это вращение усиливает смешивание с воздухом и вызывает появление завихрений, которые мы можем наблюдать при смешивании газов. Согласно [42], следующим важным фактором, влияющим на движение газа является температура. Как только струя пара была выпущена, она поднимается вверх, и более горячие участки поднимаются быстрее, чем области, которые смешались с более холодным воздухом. В то время, как газ поднимается, он испытывает внутреннее сопротивление воздуха, которое создает еще больше турбулентного вращения. Этот эффект известен под названием *термальной плавучести* (thermal buoyancy). Влияние сопротивления воздуха и термальной плавучести на движение газов приведено на рисунке 2.3.

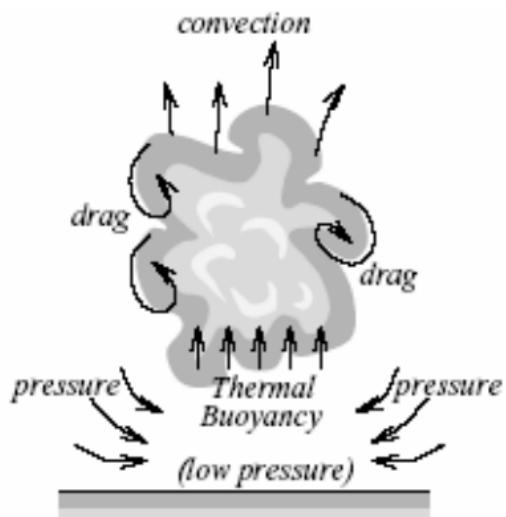


Рисунок 2.3 – Движение горячего газа

Наконец, дым и сажа появляются при недостатке в ядре огня кислорода для поддержания процесса горения. На выходе получается неполное сгорание, которое производит такие побочные продукты как сажа. Дым является результатом смешивания сажи с водой, что обычно происходит в процессе горения. Дым от огня зачастую очень темный из-за высокого содержания углекислого газа в воздухе. Однако, из-за влияния различных компонентов горючего материала или других побочных продуктов горения, в воздух могут подыматься примеси и других цветов. Например, пар имеет серый цвет (количество воды

преобладает над количеством сажи).

2.3. Особенности рендеринга в реальном времени

В [43] приводится определение системы реального времени как "системы, которой необходимо отвечать на внешние входные данные за конечный и определенный период." Таким образом, симуляция огня в режиме реального времени — это симуляция, поведение и окружение которой должно быть рассчитано за период времени, соизмеримый с ее поведением и окружением в реальном мире. Это отличается от конвейера, используемого в оффлайн симуляции при производстве фильмов, где расчет каждого кадра может занимать минуты либо часы.

В общем случае, в компьютерной графике существует большой отрыв между скоростью и реализмом. В приложениях реального времени наибольший приоритет отдается скорости. Таким образом, ключевой проблемой симуляции в реальном времени является поиск баланса между количеством вычислений для достижения оптимального результата, не допуская при этом падения частоты кадров ниже минимальной допустимой границы.

Для каждого желаемого атрибута необходимо выбрать оптимальный алгоритм для конкретного случая. Такая оптимальность зачастую достигается внедрением разнообразных ухищрений, которые имеют слабую связь с поведением объектов в реальном мире, однако они позволяют достичь желаемого эффекта. Некоторые примеры таких ухищрений будут описаны ниже в этом разделе. Суть идеи в том, что пока симуляция показывает визуально приемлемые результаты и выполняется с достаточно быстрой скоростью, скорее всего никто не заметит особых различий. Использование таких ухищрений не является чем-то зазорным, поскольку они с самого присутствуют в индустрии компьютерной графики с самого момента ее появления. По факту, на момент своего появления вся индустрия компьютерной графики сплошь состояла из таких ухищрений.

В общем случае задача симуляции огня может быть разбита на три непересекающихся подзадачи [44]:

- моделирование;
- анимация;
- визуализация.

В первую очередь необходимо выбрать подходящую внутреннюю структуру или модель для симуляции. Далее, требуется выбрать способ анимации — метод, с помощью которого будет происходить взаимодействие с моделью. Техника анимации служит для того, чтобы оживить модель, привести ее в движение. Наконец, модель и ее анимацию необходимо отрисовать на экране, используя для этого некоторые примитивы визуализации (полигоны, текстуры,

сфера, voxели и т.п.).

Альтернативная схема была предложена в [30], в которой авторы делают акцент на алгоритмах распространения огня. Фаза моделирования в данном случае является одним из этапов при разработке алгоритмов распространения огня. Схема, предложенная в данной работе, представлена на рисунке 2.1.

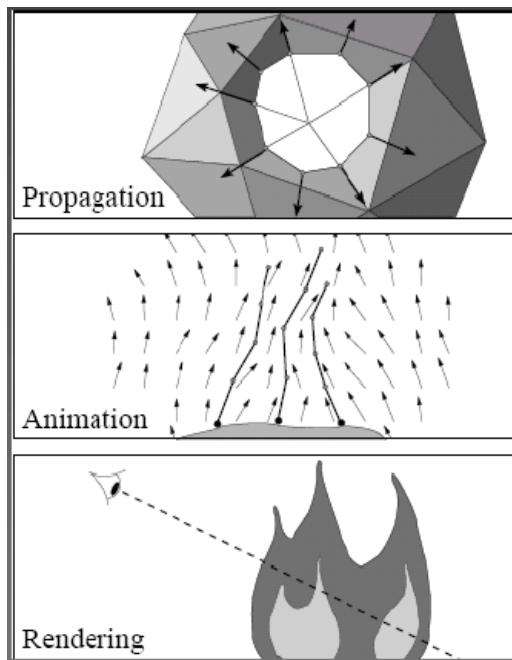


Рисунок 2.1 – Структура симуляции, предложенная в [30]

Более подробный обзор каждой стадии симуляции будет основан на схеме, предложенной в [30]. В последующих подразделах будут последовательно рассмотрены каждая из стадий симуляции, и будут описаны нюансы их взаимодействия.

2.3.1. Моделирование и визуализация огня

Стochasticеские методы моделирования огня, такие как поля турбулентности и шумов очень эффективны при создании реалистичного огня в 2D графике, но являются крайне неэффективными по количеству операций для использования в 3D анимациях в реальном времени, независимо от выбранной техники визуализации. Альтернативой может быть эффективно реализованная объемная модель рендеринга, использующая voxeli в качестве примитива визуализации, помогает достичь приемлемую для интерактивных приложений частоту кадров. Однако, при использовании данной модели сложно добиться реалистичных результатов, поскольку поверхность огня не имеет четких границ. Наиболее быстрым методом является объемное моделирование с использованием полигонов для визуализации; отрисовка полигонов происходит крайне быстро, однако полигоны не лучшее средство для визуализации языков пламени, вихрящегося дыма и частиц сажи, и поэтому обычно полигоны

пораждают довольно грубые и низкокачественные результаты. Где-то между ними находится моделирование с помощью систем частиц, данный метод может работать с желаемой скоростью, в зависимости от выбранного масштаба, который выбирается из расчета необходимого уровня детализации и выбранных техник анимации и визуализации. Как было описано в обзоре литературы, большинство симуляций огня используют системы частиц того или иного вида. Этот метод также используется в практической части. Поэтому дальнейшее описание симуляции будет сфокусировано на применении систем частиц. Системы частиц позволяют моделировать трехмерное поведение огня в интуитивной и простой манере, однако, они имеют свой набор недостатков, которые будут описаны ниже.

Одной из проблем, связанных с системами частиц, является то, что они предполагают одно и то же окружение для всех частиц в системе. Решением этой проблемы является разбиение окружения на более мелкие части, называемые *полями* [40]. Поле содержит информацию, касательно определенного участка сцены, который зачастую представляет форму куба или прямоугольной призмы, либо любой другой формы, которая может быть использована для создания мозаики, описывающей окружение. Однако, поля оказывают большую нагрузку на оперативную память. В идеальном случае, решетка должна охватывать всю сцену, которая может быть подвержена влиянию огня, при этом ячейки должны иметь достаточно малый размер, чтобы оказывать достаточно точное влияние на симуляцию. Это является крайне сложной задачей для поиска оптимального решения, т.к. установка всего лишь 10 кубов в одном из измерений кубической сцены приводит к $10^3 = 1000$ полей в сцене, при этом каждый кадр необходимо производить расчет каждого поля. При этом, данная цифра является довольно заниженной оценкой, поскольку окружение симуляции огня в основном имеет форму отличную от кубической, и поэтому высота зачастую значительно превышает размеры ширины либо глубины.

Второй проблемой систем частиц является то, что они требуют наличие огромного количества частиц для полной симуляции системы, т.к. в идеальном случае необходимо заниматься моделированием взаимодействий на молекулярном уровне. Понятно, что задачу необходимо обобщить. *Текстурный маппинг* является одной из форм такого обобщения, и некоторые варианты данного метода является часто используемым способом для достижения частоты кадров, необходимой при симуляции в режиме реального времени. Пример текстурных карт представлен на рисунке 2.2. В методе текстурного маппинга статическая анимация огня накладывается на полигоны, которые находятся на месте частиц в системе. Суть такого маппинга в том, что теперь одна частица отображает целый кластер частиц, расположенных, как на текстуре. Полигон обычно направлен на зрителя, когда тот перемещается по сцене, с помощью

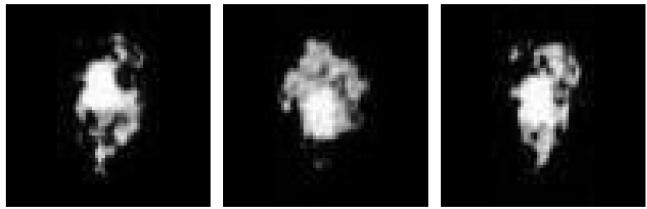


Рисунок 2.2 – Текстурные сплэты, созданные на основе фотографий [18]

вращения вдоль оси y . Однако, как можно заметить, в таком случае полигон может быть направлен на зрителя только тогда, когда зритель передвигается вдоль осей x и y , и, как можно предположить, огонь будет выглядеть довольно странно, если смотреть на него сверху, особенно сильно это будет заметно, если посмотреть на сцену под острым углом. Еще одной проблемой, связанной с использованием текстурного маппинга, является то, что освещение сцены будет считать полигоны плоской поверхностью, и применять соответствующие эффекты освещения к огню, которые, конечно, будут выглядеть крайне нереалистично.

Одним из улучшений текстурного маппинга является текстурный сплэттинг, который является техникой для симуляции объемного рендеринга. В текстурном сплэттинге битовая карта складывается с картой прозрачности, таким образом битовая карта становится частично прозрачной. Другими словами, текстурные приближения на разных уровнях накладываются друг на друга с использованием смешивания.

Другим улучшением текстурного маппинга является использование последовательностей изображений реального огня, вместо постоянного переиспользования одной и той же текстуры для каждого полигона. Данное улучшение помогает улучшить визуальное восприятие сцены с помощью уменьшения количества статических элементов. Эта техника хорошо себя зарекомендовала и позволяет получать довольно убедительные результаты.

Как можно заметить, метод текстурного маппинга позволяет успешно встраивать заранее отрисованные элементы в симуляции в режиме реального времени. Так же можно заметить, что при встраивании 2D текстур в 3D симуляцию, симуляцию уже сложно назвать по-настоящему трехмерной. Однако, это является одним из тех ухищрений, про которые шла речь в начале этого раздела. У текстур нет аналога в реальном мире, поэтому это является своеобразным ухищрением. Текстуры слегка смывают различие между 2D и 3D симуляцией, однако, использование текстур 2D текстур в трехмерной симуляции довольно популярно, поскольку позволяет существенно снизить вычислительную нагрузку. Разработанная в ходе исследования система также использует данный прием.

2.3.2. Анимация огня

Существует множество способов анимации огня. Среди них можно выделить подходы, использующие дифференциальные уравнения. Данные подходы позволяют получить точные анимации, однако, они требуют отслеживания огромного числа переменных. Производимые с помощью данных подходов симуляции тяжело контролировать, поскольку их параметрами являются физические константы, чью взаимосвязь с желаемыми визуальными характеристиками зачастую крайне сложно установить. Вдобавок значения для этих параметров необходимо подбирать вручную, поскольку природа огня еще до конца не изучена. Наиболее важно то, что эти подходы требуют выполнения значительного количества вычислений, и поэтому их сложно применять в симуляции в режиме реального времени.

Решением данной проблемы может быть попытка реализовать лишь наиболее важные физические свойства огня, которые описаны в разделе 2.2.4. Далее необходимо добиваться максимально возможной реалистичности и эффектности каждого из компонентов. При этом следует уделить внимание тому, чтобы реализация выглядела достаточно правдоподобно и не была слишком уж заточена под конкретную сцену. Необходимо стремиться к более элегантным решениям.

Выводы по главе 2:

1. Для разработки практической части диссертации будет использована библиотека OpenGL. Core профиль OpenGL предоставляет большую свободу действий, что дает возможность тщательно реализовать и настроить различные атрибуты огня. Использование игровых движков является чрезмерным для данной задачи.

2. В практической части диссертации будет выполняться симуляция свободно питаемого огня. Примерами такого огня в реальной жизни могут служить факел либо костер. Данный вид огня представляет интересную задачу для разработки анимации, и процессы, лежащие в его основе лучше поддаются математическому описанию, чем процессы, протекающие в сильном огне.

3. Системы частиц будут использованы для моделирования огня. Для устранения недостатков данного метода будут применены различные приемы, такие как текстурный маппинг, рандомизация размеров, скорости и количества частиц.

4. Анимация огня с помощью систем дифференциальных уравнений предъявляет высокие требования к аппаратным ресурсам и крайне сложно конфигурируется. Для создания анимации в рамках диссертации будут использованы более простые модели. Основной упор будет сделан на визуальное восприятие сцены.

ГЛАВА 3

РАЗРАБОТКА СИСТЕМЫ ДИНАМИЧЕСКОЙ СИМУЛЯЦИИ ОГНЯ

Фокус данного исследования направлен на создание реалистично выглядящего огня, отрисовка которого происходит в режиме реального времени. Это включает в себя создание компьютерной программы, которая служит для изучения и моделирования внешнего вида и поведения огня, и при этом плавно работает в режиме реального времени. В предложенной реализации моделируется поведение костра. Для моделирования огня была выбрана система частиц, предложенная в [2].

Моделирование огня в режиме реального времени можно условно разделить на научное и эстетическое. Несмотря на то, что обе модели стремятся создать реалистичное пламя, цель и практическое приложение данных моделей отличается. Научные модели нацелены на исследование динамики определенного феномена. Эти модели нацелены на точное представление огня с учетом лежащей в основе поведения огня физики, включая термодинамику. Целью научных моделей является получение более глубоких знаний о поведении огня. Научные модели часто используются в экспериментах, изучении экологических проблем и проблем окружающей среды, военных симуляторах.

Напротив, эстетические модели более сосредоточены на визуальном представлении огня, то есть на внешнем виде пламени на экране. Целью эстетических моделей является воссоздание визуальных эффектов, присущих огню, используя при этом сравнительно менее ресурсозатратные техники и методики расчетов по сравнению с научными моделями. Таким образом, эстетические модели лучше подходят для использования на компьютерах с ограниченными вычислительными ресурсами а также для использования в приложениях реального времени, которые крайне чувствительны к задержкам и падению производительности. Примером таких приложений могут служить видеоигры.

Модель огня костра, разработанную в ходе данного исследования, можно отнести к эстетической. Основное внимание при разработке системы уделялось воссозданию поведения и внешнего вида огня, близких к их аналогам в реальном мире. При создании симуляции использовались некоторые ухищрения, упрощения и ограничения, которые имеют мало общего с физическими процессами, протекающими в реальном огне.

3.1. Моделирование огня

При моделировании динамического поведения огня с помощью системы частиц визуальное восприятие сцены сильно зависит от четырех важных атрибутов частиц: формы, светимости, прозрачности и цвета. Общая форма огня зависит от размера частиц и их анимации. Такие эффекты, как мерцание и разделение пламени, зависят от динамического поведения частиц. Светимость частиц используется для симуляции эффекта накаливания в пламени. Накаливание — это эффект излучения нагретыми телами света. Пламя представляет собой газовый феномен, который имеет высокую температуру и выделяет большое количество энергии в окружающую среду. Таким образом, сквозь пламя могут просвечиваться объекты, находящиеся по другую сторону. Степень прозрачности частиц позволяет моделировать эффект полупрозрачности, наблюдаемый в реальном пламени.

В данной работе для воссоздания цвета и внешнего вида огня были использованы текстуры. Наблюдения показывают, что цвет пламени зависит от типа топлива и вида окислителя, которые взаимодействуют в процессе горения. Углеродное топливо в основном порождает языки пламени желтого либо оранжевого цветов, в то время как различные газы порождают языки пламени голубоватого оттенка. Для упрощения задачи в рамках исследования производилось моделирование только углеродного топлива. Далее будет приведено описание разработанной модели на основе системы частиц.

Разработанная система имеет три ключевых элемента: частицы, эмиттеры, менеджеры эмиттеров. Данная структура основана на схеме, предложенной в [11]. Частицы представлены как небольшие объекты без четко определенного размера, структуры и траектории движения. Данная недерминистичная природа частиц была достигнута путем использования стохастических процессов для вычисления значения атрибутов эмиттера и частиц. Данное решение позволило достаточно убедительно смоделировать хаотическую природу огня. Общая схема алгоритма основана на схеме, предложенной Вудхаузом в [45] (рис. 3.1).

Частицы создаются эмиттерами и затем вводятся в систему частиц. Эмиттеры являются случайными точками, расположенными в рамках указанных границ. По факту, можно считать эмиттеры статическими частицами. Эмиттеры обладают различными характеристиками, которые вносят вклад в динамику частиц и общий внешний вид модели огня. Ключевыми атрибутами эмиттеров являются их позиция, скорость эмиссии частиц, начальный вектор эмиссии, динамический список частиц и энергия эмиттера.

Местоположение эмиттеров определяет, где в пределах системы частиц будут создаваться эмиттеры. В предложенной реализации эмиттеры создаются внутри круга, который обозначает радиус огня. Эмиттеры также управляют скоростью эмиссии частиц и преобладающим направлением эмиссии. Ско-

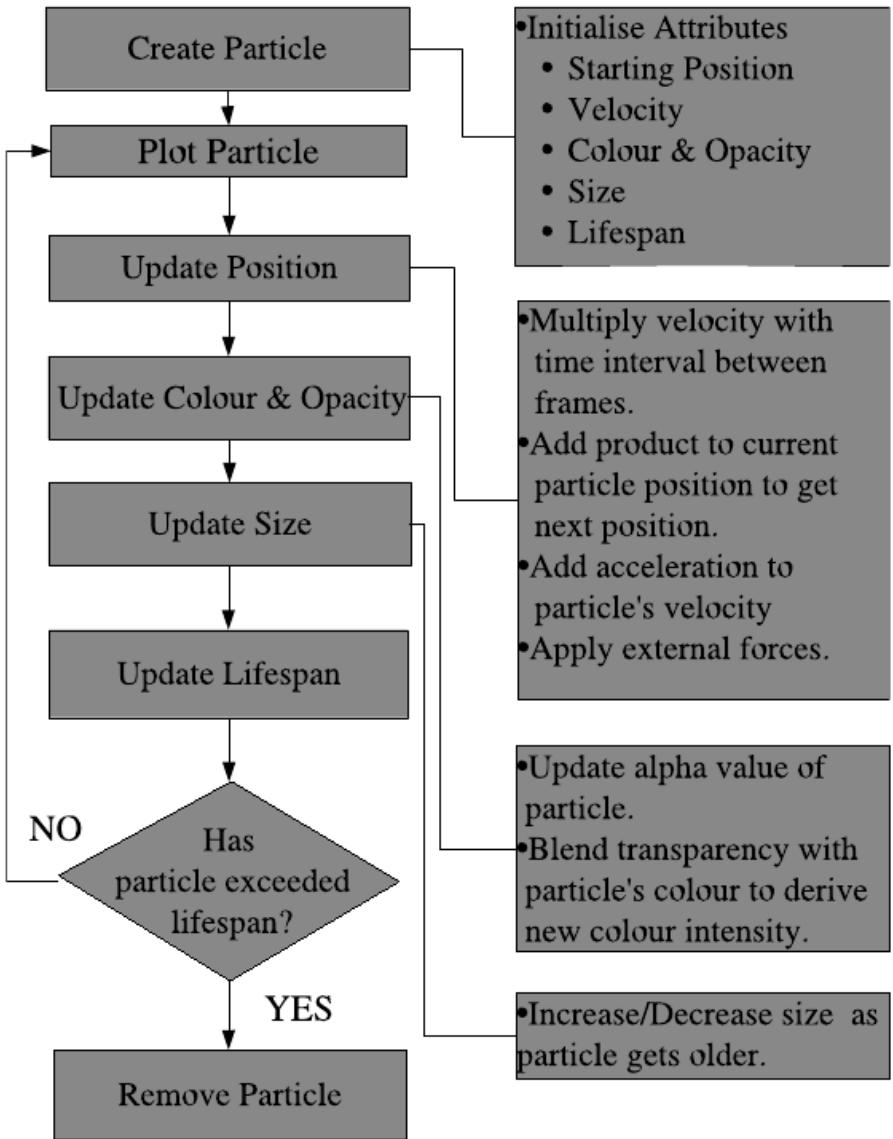


Рисунок 3.1 – Схема обновления частиц в кадре, предложенная Вудхаузом в [45]

рость эмиссии или скорость горения определяет число частиц, которые испускает каждый эмиттер в момент времени. Как видно на рисунке 3.2 частицы могут появляться в виде слоев на экране. Это происходит потому, что все частицы с постоянной скоростью генерируются у основания эмиттера, вектора движения частиц также совпадают. Для устранения данного недостатка можно при генерации частиц добавить им случайное смещение вдоль оси y , также можно добавить небольшие случайные смещения к начальным векторам скорости частиц. Доработанная система демонстрирует более реалистичные результаты и представлена на рисунке 3.3. Эффект кластеризации может приводить к видимым разрывам в анимации из-за существенного перепада количества частиц. Оптимальные значения для коэффициентов смещения следует подбирать опытным путем.

Когда эмиттер создается ему присваивается случайное количество частиц, которые хранятся в динамическом списке. Список изменяется при до-

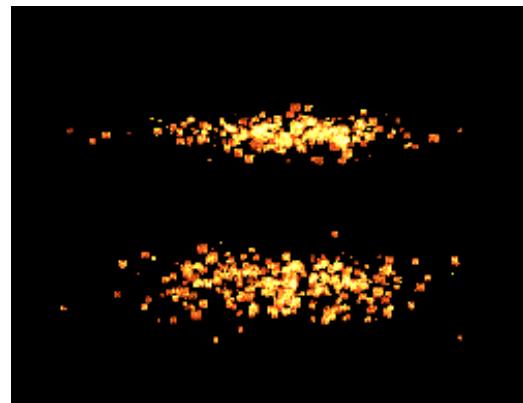


Рисунок 3.2 – На картинке представлен эффект появления слоев, состоящих из частиц испущенных в один момент времени

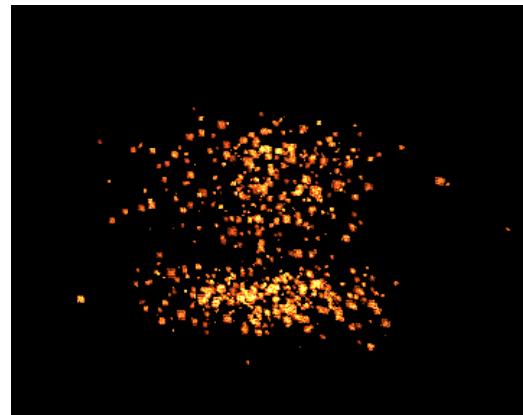


Рисунок 3.3 – Уменьшение кластеризации частиц с помощью добавления случайных смещений

бавлении или удалении частиц из системы. Как можно понять, производительность данного решения крайне страдает из-за постоянных операций по управлению памятью. Решение данной проблемы взято из [35]. В данной работе предложено выделять память заранее для каждого эмиттера, выделенная область ограничивает максимальное количество частиц присутствующих в системе, которые были выделены одним эмиттером. Для обновления атрибутов и дальнейшего рендеринга выбираются только те частицы, которые имеют положительное значение оставшегося времени жизни. При добавлении новой частицы в систему происходит поиск в списке уже неактивной частицы и замена ее на новую.

Недостатками данной реализации являются:

1. Необходимость определения максимального количества частиц в системе заранее. Это несет существенные затраты по памяти. Также при отсутствии свободных частиц происходит замена случайной частицы на новую, что может быть нежелательно.
2. Высокая алгоритмическая сложность алгоритма поиска новой частицы. Алгоритм имеет линейную сложность ($O(n)$) в худшем случае. В алгоритме есть небольшая оптимизация. Алгоритм хранит индекс последней неис-

пользованной частицы, и пытается на следующем шаге начинать поиск с нее. Эта оптимизация слабо помогает при постоянном наличии большого числа частиц в системе. Если у частиц еще значительно варьируется время жизни — практически всегда выполняется линейный поиск.

Наибольшую проблему вызывал алгоритм обновления частиц. Он являлся бутылочным горлышком системы. Для решения данной проблемы было разработано тривиальное, но эффективное решение: в цикле обновления частиц индексы "умерших" частиц помещаются в отдельный список. Данный список используется на следующем шаге симуляции на этапе генерации новых частиц. Данное решение немного увеличило затраты по памяти, однако позволило выполнять операцию поиска "умерших" частиц за константное время ($O(1)$).

Вернемся к описанию эмиттеров. Энергия эмиттера определяет время жизни системы частиц. Каждую итерацию энергия эмиттера уменьшается, пока не достигнет минимального значения, обозначающего, что эмиттер иссяк. Иссякшие эмиттеры убираются из системы вместе со всеми назначеными им частицами. Менеджер эмиттеров содержит список всех эмиттеров в системе. Когда эмиттер создается, он добавляется в динамический список. Менеджер эмиттеров каждую итерацию обновляет все эмиттеры и удаляет иссякшие эмиттеры из системы. В таблице 3.1 приведено описание всех атрибутов эмиттеров, описанных на данный момент.

Таблица 3.1 – Атрибуты эмиттера

Атрибут	Описание
Позиция	Местоположение эмиттера в трехмерном пространстве
Энергия	Время жизни эмиттера
Скорость	Модуль начальной скорости испускаемых эмиттером частиц
Направление	Единичный вектор, указывающий направление эмиссии частиц
Радиус	Радиус эмиттера. Ограничивает область генерации частиц
Список частиц	Динамически обновляемый список принадлежащих эмиттеру частиц

В качестве структуры симулятора была выбрана схема, предложенная в [11]. Общая структура симулятора представлена на рисунке 3.4.

При описании структуры симулятора, достаточного внимание не было уделено описанию частиц. Частицы являются ключевыми объектами в симуляции. Они имеют такие атрибуты как позицию, энергию (время жизни), цвет, скорость, ускорение и размер. Каждая частица появляется в системе недалеко от центра эмиттера, в дальнейшем ее местоположение изменяется в течение всего времени жизни частицы. Скорость частицы определяет ее следующее местоположение в системе. Для улучшения визуального восприятия сцены, к частицам также применяется ускорение. Как и эмиттеры, частицы также обладают атрибутом энергии, который определяет время жизни частицы в системе.

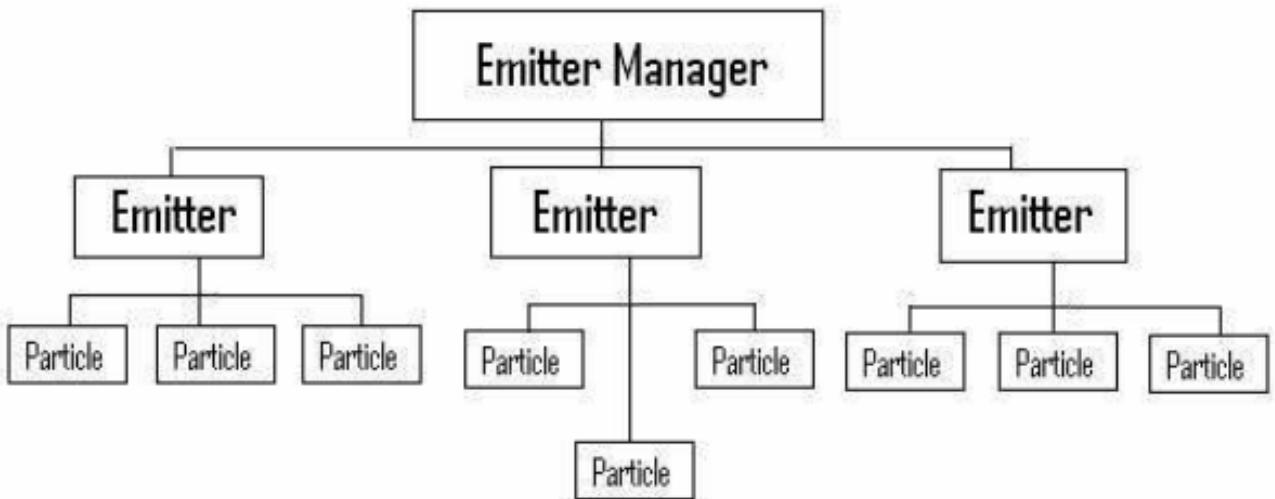


Рисунок 3.4 – Иерархия объектов, использованная в разработанном симуляторе

Когда частица исчерпала свою энергию, ее место занимает новая, что позволяет оптимизировать операции по управлению памятью. Цвет и размер частицы определяется на основе количества энергии частицы и играет существенную роль в формировании визуального представления сцены. По мере уменьшения энергии частицы ее размер, интенсивность цвета и прозрачность уменьшаются. Это помогает создать эффект незаметного исчезновения частиц по истечении их времени жизни. Краткий обзор всех атрибутов частиц представлен в таблице 3.2.

Таблица 3.2 – Атрибуты частицы

Атрибут	Описание
Позиция	Местоположение частицы в трехмерном пространстве
Скорость	Вектор скорости частицы
Ускорение	Вектор ускорения частицы
Энергия	Представляет оставшееся время жизни частицы
Цвет	Цвет частицы. Интенсивность и прозрачность уменьшаются по мере старения частицы
Размер	Размер частицы. Размер частицы уменьшается по мере старения частицы

Первый прототип системы представлен на рисунке 3.5. Данный прототип страдал от множества недостатков:

1. Высокая загрузка ГП. Несмотря на сравнительно небольшое количество объектов загрузка ГП достигала 85%. При этом в системе находилось всего лишь 500 частиц. Каждый кадр генерировалось по 20 частиц.
2. Проблемы с текстурами. Ошибка с наложением текстур была исправлена в следующих версиях.
3. Отсутствие должной анимации частиц. Все частицы летят строго в направлении эмиссии. Созданию анимации пламени будет посвящен отдельный раздел.

Ключевой проблемой на данной стадии являлась чрезмерная загрузка

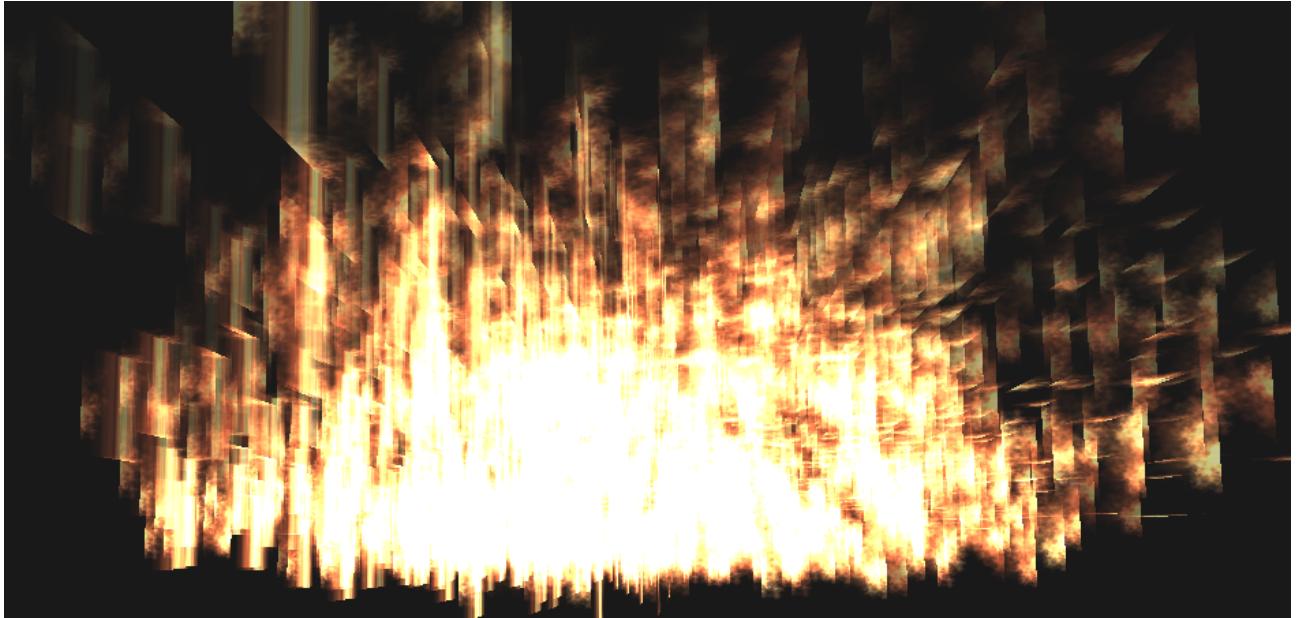


Рисунок 3.5 – Первый прототип системы

ГП. Для устранения этой проблемы была задействована техника инстансинга. Идеей данного метода является уменьшение количества вызовов к ГП. Более подробно эта техника описана в [33, 35]. Данная техника хорошо подходит для рендеринга множества однородных объектов, такими и являются частицы. Вместо того, чтобы в цикле выставлять атрибуты частицы и делать вызов для функции отрисовки частицы, атрибуты всех частиц записываются в массивы. Рендеринг всех частиц осуществляется в помощь одного обращения к ГП. Реализация данной техники позволило увеличить количество частиц в системе до 5000, количество частиц создаваемых за кадр — до 750. При этом нагрузка на ГП снизилась до 28%. Модифицированная система представлена на рисунке 3.6.

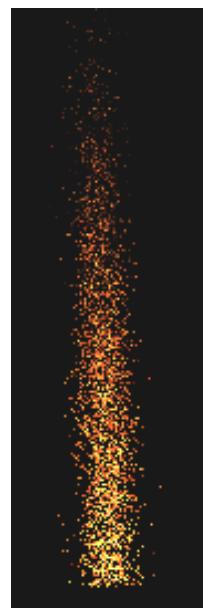


Рисунок 3.6 – Использование инстансинга для отрисовки частиц

Увеличение максимального числа частиц и уменьшение их размера положительно скажется на реалистичности анимации. Однако, оптимизация работы ГП продемонстрировала недостатки алгоритмов, работающих на ЦП. Загрузка ЦП достигала 100% (симуляция выполняется в один поток).

Для решения данной проблемы был использована оптимизация алгоритма обновления частиц, описанная выше в данном разделе. Данная оптимизация позволила снизить среднюю загрузку ЦП до 18%, что позволило выполнять симуляцию с частотой 60 кадров / секунду. Изменения можно увидеть на рисунке 3.7.

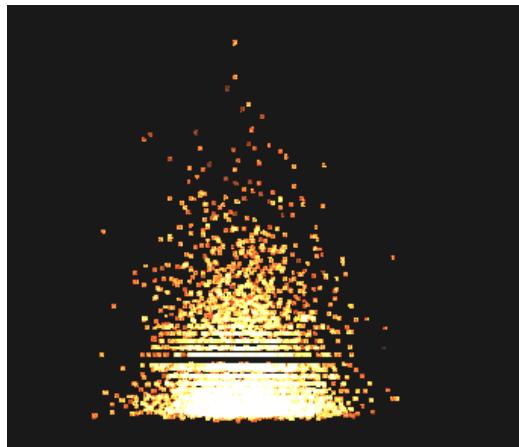


Рисунок 3.7 – Система, основанная на новом алгоритме поиска ”мертвых” частиц

Изменение высоты пламени и скорости движения частиц вызваны изменением алгоритма поиска ”мертвых” частиц. В прошлой версии алгоритма при отсутствии свободных частиц добавления новой частицы не происходило, в новой версии происходит замена случайной частицы при неудаче. В продемонстрированном примере часто происходит удаление случайных частиц и замена их на новые. Из-за данного эффекта увеличилась плотность частиц у основания пламени, визуально увеличилась скорость частиц. Удаление случайных частиц позволило устраниТЬ эффект непрерывного столба пламени, наблюдаемый на рисунке 3.6. Данный эффект положительно сказался на визуальном восприятии сцены и будет использован в дальнейшем.

Дальнейшие улучшения были связаны с увеличением стохастичности симуляции. К значениям многих атрибутов были добавлены случайные смещения, что позволило уменьшить количество статических элементов симуляции. Система с внесенными оптимизациями и улучшениями представлена на рисунке 3.8. Однако, наиболее заметным недостатком данной системы является отсутствие анимации. Решение, использованное для анимации системы будет описано в следующем разделе.

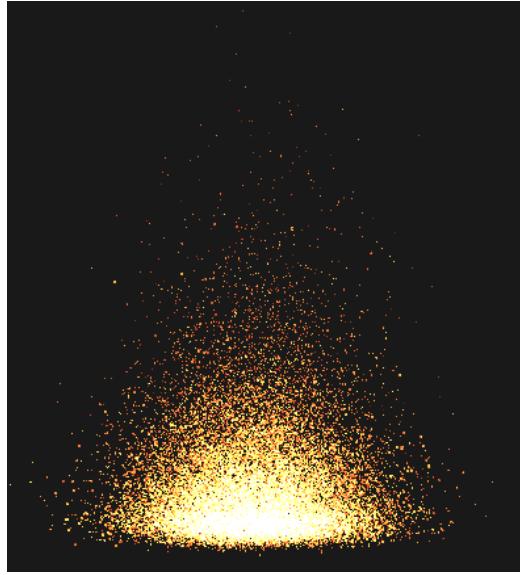


Рисунок 3.8 – Использование стохастических эффектов в симуляции

3.2. Анимация пламени

В ходе исследования различных алгоритмов для анимации огня, был проведен обзор нескольких различных подходов, описание которых можно найти в главе 1. В общем случае, в момент времени t новое положение частицы зависит от ее скорости $\vec{v}(t)$, ускорения \vec{a} , и интервала времени между кадрами Δt . Как было замечено ранее, начальный вектор скорость частицы определяется вектором эмиссии конкретного эмиттера. Для расчета следующего положения частицы $\vec{p}(t + \Delta t)$, используется следующие уравнения (3.1, 3.2).

$$\vec{p}(t + \Delta t) = \vec{p}(t) + \vec{v}(t) \cdot \Delta t \quad (3.1)$$

$$\vec{v}(t + \Delta t) = \vec{v}(t) + \vec{a} \quad (3.2)$$

где $\vec{p}(t + \Delta t)$ — координаты частицы в следующий момент времени;

$\vec{p}(t)$ — текущие координаты частицы;

$\vec{v}(t)$ — скорость частицы в текущий момент времени;

Δt — величина временного интервала между кадрами;

$\vec{v}(t + \Delta t)$ — скорость частицы в следующий момент времени;

\vec{a} — текущее ускорение частицы.

Как видно из уравнений выше, движение частиц в системе является равнотускренным. Частицы немного ускоряются по мере их движения. Для данной симуляции была эмпирически подобрана величина ускорения, вычисляемая по следующей формуле (3.3).

$$\vec{a} = 0,02 \cdot \vec{v}_0 \quad (3.3)$$

где \vec{a} — ускорение частицы;

\vec{v}_0 — начальная скорость частицы.

Как можно заметить, описанные выше уравнения описывают движение частиц как равноускоренное прямолинейное движение. Однако, как было описано в разделе 2.2.4, воздействующие на огонь силы создают более сложное движение частиц пламени. Важным отличием реального огня, от описанной выше модели, является постоянное движение и изменение формы языков пламени. В статье [46] Том Харрис дает описание этому феномену. Данный эффект происходит из-за того, что когда газы нагреваются, они становятся менее плотными, чем окружающий их воздух. Поэтому газы двигаются в зону, где давление ниже. Чтобы просимулировать данный эффект в объеме, покрывающем зону горения, случайным образом выбираются точки. Эти случайные точки симулируют зоны с низким давлением. Каждую итерацию эти точки заменяются на другие, также выбранные случайным образом.

Когда частица попадает в систему, она движется в сторону ближайшей точки с низким давлением. Таким образом, точки с низким давлением формируют острые языки пламени, которые можно наблюдать в реальном мире. Данный алгоритм представлен на рисунке 3.1. Реализация данного алгоритма представлена на рисунке 3.2.

- – Random Pressure Points
- – Particles
- – Flow of Particles

- – Random Pressure Points
- – Particles
- – Flow of Particles

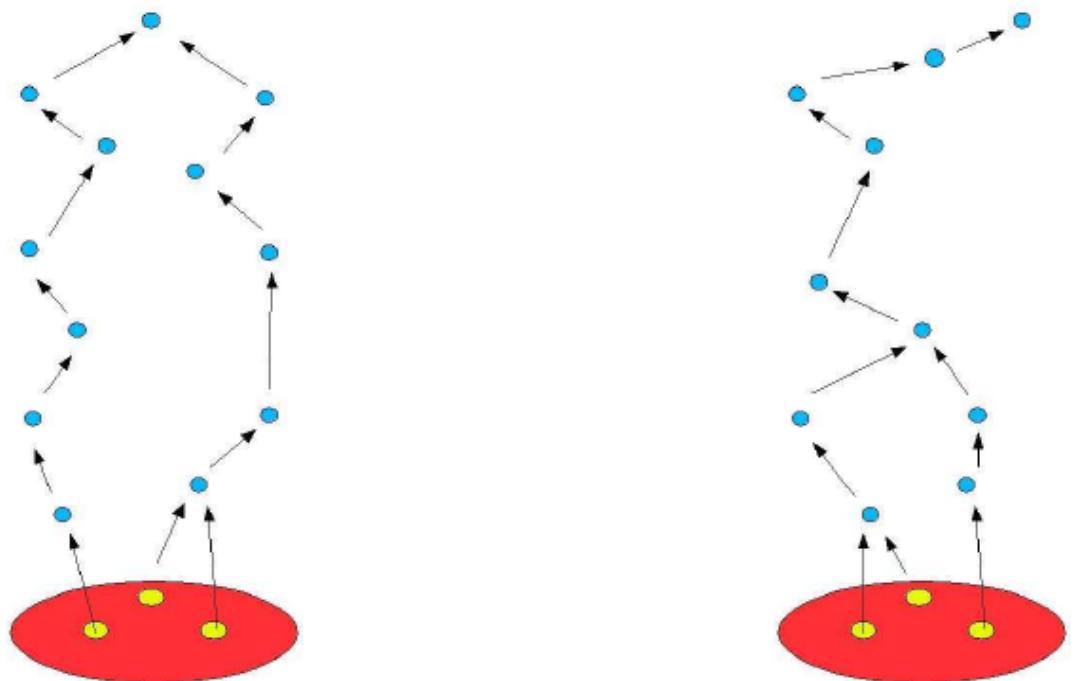


Рисунок 3.1 – Алгоритм анимации частиц

Одной из сложностей при реализации данного алгоритма была реализация быстрого алгоритма поиска ближайшей к частицы точки с низким дав-

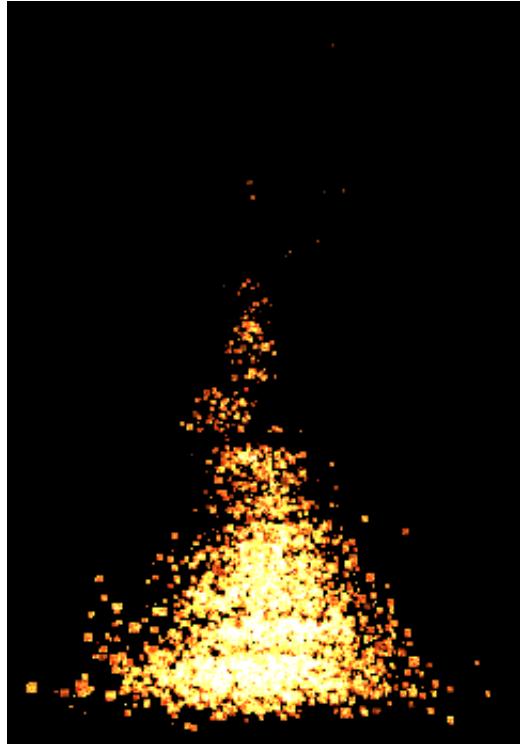


Рисунок 3.2 – Реализация анимации частиц

лением. В версии на рисунке 3.2 используется 5000 частиц (m) и 500 точек низкого давления (n). Линейный поиск по такой системе имеет сложность $O(m) \cdot O(n)$. В оптимизированном методе используется алгоритм быстрой сортировки ($O(n \log n)$) для упорядочивания точек низкого давления по величине координаты y , в отсортированном массиве выполняется бинарный поиск ($O(\log_2 n)$) ближайшей к частице точки низкого давления (сравнение по расстоянию до точки и значению координаты y). Поиск ближайшей точки низкого давления по новому алгоритму имеет сложность $O(n \log n) + O(m) \cdot O(\log_2 n)$.

3.3. Результаты экспериментов

Компьютерная симуляция, описанная ранее в этой главе, была успешно разработана и реализована на основе кодовой базы из примеров, предложенных в [35]. Разработанный симулятор написан на языке C++ с использованием программного интерфейса OpenGL 4.5. Программа была скомпилирована с помощью компилятора GNU G++ версии 8.3.0. Окружение симуляции использует операционную систему Debian 10 Buster, процессор Intel Core i5-5200U с частотой 2.7GHz, 8 ГБ оперативной памяти и видеокарту Intel(R) HD Graphics 5500.

В анимации изображения быстро сменяют друг друга на экране, при этом их содержание изменяется незначительно. Данный эффект используется для того, чтобы заставить человеческий глаз воспринимать изменения как перемещение объектов. Чем выше частота кадров, тем плавнее воспринимается дви-

жение объектов, в то время как при низкой частоте кадров перемещения выглядят дергано. В фильмах и телевидении используется частота в 24 кадра/секунду и 30 кадров/секунду соответственно. Высокая частота кадров крайне важна в видеоиграх, где частота кадров сильно влияет на игровой процесс. В консольных видеоиграх средняя частота кадров составляет 30 кадров в секунду, рекомендуемой частотой ПК игр является 60 кадров в секунду.

Одним из важных экспериментов является поиск максимального количества частиц в системе, при котором сохраняется приемлемая частота кадров. Для данного теста было выполнено несколько опытов, в ходе которых максимальное количество частиц в системе принимало значения в 5000, 12500, 25000, 50000 частиц. Результаты представлены в таблице 3.1.

Таблица 3.1 – Зависимость частоты кадров от количества частиц в системе

Количество частиц	Средняя частота кадров
5000	60,00
10000	58,46
15000	50,62
25000	31,94
50000	15,85

Как видно из таблицы 3.1, при 25000 частиц в системе симуляция работает с еще приемлемой частотой кадров. Измерение нагрузки ЦП и ГП при моделировании 25000 частиц показало всего лишь 26% загрузку ГП и 100% загрузку ЦП. Таким образом, у системы имеется потенциал для увеличения производительности, однако, для этого требуется выполнить оптимизацию алгоритмов, работающих на ЦП.

Выводы по главе 3:

1. Реализованная система имеет множество оптимизаций для улучшения производительности, что позволяет выполнять симуляцию нескольких тысяч частиц в системе в реальном времени даже на относительно слабом аппаратном обеспечении.
2. Разработанный алгоритм анимации занимает мало вычислительных ресурсов, однако предоставляет ограниченную реалистичность. Отсутствуют такие вещи как отделение участков пламени и эффекты турбулентности. Также сложно вручную настроить анимацию пламени.
3. В предложенном модели отсутствуют алгоритмы распространения огня, механизмы для взаимодействия с окружающей средой. Данные проблемы могут стать предметом дальнейших исследований.

ЗАКЛЮЧЕНИЕ

В ходе исследования предмета диссертации был проведен обзор большого количества работ по данной теме. Анализ литературных источников показал наличие множества кроссдисциплинарных связей в решениях данной проблемы. Поведение реального огня описывается физикой, в том числе термодинамикой, создание симулятора тесно связано с математическим моделированием и компьютерной графикой. Также встречаются и менее очевидные решения, которые позволяют добиться интересных результатов за счет использования методов и приемов из, на первый взгляд, не связанных дисциплин. Например, использование алгоритма косяка птиц из теории роевого поведения позволило создать согласованное движение частиц в языке пламени [11]. Таким образом симуляция огня является сложной комплексной задачей, которая в данный момент не может быть решена полностью. Для успешного решения задачи симуляции огня необходима разработка специализированных решений, сфокусированных на моделировании ограниченного числа атрибутов огня.

Данная работа направлена на создание трехмерной симуляции огня, которая может быть использована в трехмерных видеоиграх. В ходе создания симулятора приоритетными задачами являются оптимизация использования вычислительных ресурсов и улучшение визуальной составляющей симуляции. Разработанная система показывает хорошую производительность, однако показывает не слишком реалистичные визуальные результаты. Для устранения недостатков визуализации могут быть использованы техники процедурного моделирования текстур, например, шум Перлина.

В дальнейших исследованиях по данной работе планируется уделить внимание алгоритмам распространения фронта огня, взаимодействию огня с окружающими объектами и окружающей средой. Основой для данных исследований могут послужить идеи, предложенные в работах [15, 17, 44].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Список публикаций соискателя

- 1–A. *Стаховский, А. В. Современные алгоритмы моделирования аморфных объектов / А. В. Стаковский // Компьютерные системы и сети: 55-я юбилейная научная конференция аспирантов, магистрантов и студентов, Минск, 22-26 апреля 2019 г. — Минск : Белорусский государственный университет информатики и радиоэлектроники, 2019. — С. 63.*
- 2–A. *Стаховский, А. В. Динамическая симуляция объемного огня / А. В. Стаковский // Компьютерные системы и сети: 56-я научная конференция аспирантов, магистрантов и студентов, Минск, 21-22 апреля 2020 г. — Минск : Белорусский государственный университет информатики и радиоэлектроники, 2020. — С. 63.*
- 3–A. *Стаховский, А. В. Анализ современных алгоритмов симуляции огня / А. В. Стаковский // Молодой ученый. — 2019. — Нояб. — № 47. — С. 100—105.*

Список использованных источников

1. *Gillies, D. F.* Third Year and MSc Interactive Computer Graphics Course / D. F. Gillies // — Chap. Lecture 17: Graphical Simulation of Fire.
2. Particle systems — a technique for modeling a class of fuzzy objects. — Detroit, Michigan, 07/1983. — Proceedings of the 10th annual conference on computer graphics and interactive techniques.
3. Physically Based Modeling and Animation of Fire. — San Antonio, Texas, 07/2002. — Proceedings of the 29th annual conference on computer graphics and interactive techniques.
4. *Lamorlette, A.* Structural Modeling of Flames for a Production Environment / A. Lamorlette, N. Foster // ACM Transactions on Graphics. — 2002. — June. — Vol. 21.
5. *Stock, M.* Smoke Water Fire / M. Stock // ACM SIGGRAPH 2008 Art Gallery. — Los Angeles, California : ACM, 2008. — P. 102–102.
6. *Feldman, B.* Animating suspended particle explosions / B. Feldman, J. O'Brien, O. Arikan // ACM Trans. Graph. — 2003. — July. — Vol. 22. — P. 708–715.
7. *Selle, A.* A vortex particle method for smoke, water and explosions / A. Selle, N. Rasmussen, R. Fedkiw // ACM Trans. Graph. — 2005. — July. — Vol. 24. — P. 910–914.
8. *Studio, T.* The remarkable visual effects in The Hobbit: The Battle of the Five Armies / T. Studio. — 02/2015. — URL: <https://www.slideshare.net/andrewpsclark/the-remarkable-visual-effects-in-the-hobbit-the-battle-of-the-five-armies-44914535> (visited on 04/23/2020).
9. *Kruijff, M. de.* firestarter – A Real-Time Fire Simulator : Master's thesis / de Kruijff Marc. — Computer Science Capstone.
10. *Green, S.* NVIDIA FlameWorks: Real-time Fire Simulation / S. Green // ACM SIGGRAPH 2014 Computer Animation Festival. — Vancouver, Canada : ACM, 2014. — P. 1–1.
11. *Somasekaran, S.* Using Particle Systems to Simulate Real-Time Fire / S. Somasekaran //. — 2005.
12. *Vanzine, Y.* Realistic Real-Time Rendering of Fire in a Production System : Feasibility Study / Y. Vanzine //. — 2007.

13. *Hladký, J.* Fire Simulation in 3D Computer Animation with Turbulence Dynamics including Fire Separation and Profile Modeling / J. Hladký, R. Ďuríkovič // International Journal of Networking and Computing. — 2018. — July. — Vol. 8. — P. 186–204.
14. Three Dimensional Fire Simulation Based on Visual Learning of Image Features / C.-N. Lee [et al.] // IEEE Visualization. — 2011.
15. Voxels on Fire / Y. Zhao [et al.] // IEEE Visualization. — 2003.
16. *Lyes, T. S.* Fire and Flame Simulation using Particle Systems and Graphical Processing Units / T. S. Lyes, K. A. Hawick //. — 2013.
17. Meshes on Fire / H. Lee [et al.] // Computer Animation and Simulation 2001. — Vienna : Springer Vienna, 2001. — P. 75–84.
18. Simulating Fire with Texture Splats. / X. Wei [et al.] //. — 01/2002. — P. 227–234.
19. *Jo, E.* Lattice-Boltzmann and Eulerian Hybrid for Solid Burning Simulation / E. Jo, B. Kim, O.-Y. Song // Symmetry. — 2019. — Vol. 11. — P. 1405.
20. A data-driven approach for synthesizing high-resolution animation of fire / S. Sato [et al.] // DigiPro '12. — 2012.
21. Physics-driven Fire Modeling from Multi-view Images / G. Dorta [et al.]. — 2018. — Apr.
22. *Stam, J.* Depicting Fire and Other Gaseous Phenomena Using Diffusion Processes / J. Stam, E. Fiume. — 2001. — June.
23. *Melek, Z.* Interactive simulation of fire / Z. Melek, J. Keyser //. — 02/2002. — P. 431–432.
24. *Bangalore, A.* A technique for art direction of physically based fire simulation / A. Bangalore, D. H. House // CAe '12. — 2012.
25. *Stam, J.* Real-Time Fluid Dynamics for Games / J. Stam. — 2003. — May.
26. *Nishita, T.* Modeling and rendering of various natural phenomena consisting of particles / T. Nishita, Y. Dobashi //. — 02/2001. — P. 149–156.
27. *Green, S.* Flame On: Real-Time Fire Simulation for Video Games / S. Green, C. Horvath. — 04/2017.
28. Real-time procedural volumetric fire / A. Fuller [et al.] //. — 01/2007. — P. 175–180.
29. *Balci, M.* Real-time 3D fire simulation using a spring-mass model / M. Balci, H. Foroosh //. Vol. 2006. — 01/2006. — 8 pp.

30. *Beaudoin, P.* Realistic and Controllable Fire Simulation. / P. Beaudoin, S. Paquet, P. Poulin // . — 01/2001. — P. 159–166.
31. Realistic Fire Simulation: A Survey. — 2011. — 12th International Conference on Computer-Aided Design and Computer Graphics.
32. *Самаль, Д. И.* Машинная графика : лаборатор. практикум для студентов специальности 1-40 02 01 «Вычисл. машины, системы и сети» / Д. И. Самаль, В. А. Супонев, В. А. Прытков. — Минск : БГУИР, 2009. — С. 43.
33. *Sellers, G.* OpenGL Superbible: Comprehensive Tutorial and Reference / G. Sellers, R. S. Wright, N. Haemel. — Addison-Wesley Professional, 2015.
34. *Group, K.* History of OpenGL / K. Group. — 02/2019. — URL: https://www.khronos.org/opengl/wiki/History_of_OpenGL (visited on 04/27/2020).
35. *Vries, J. de.* Learn OpenGL: An offline transcript of learnopengl.com / J. de Vries. — 06/2017.
36. Пламя. — URL: <https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%BC%D1%8F> (дата обр. 28.04.2020).
37. СТ СЭВ 383-87. Пожарная безопасность в строительстве. Термины и определения. — 07.1987.
38. Огонь. — URL: <https://ru.wikipedia.org/wiki/%D0%9E%D0%B3%D0%BE%D0%BD%D1%8C> (дата обр. 28.04.2020).
39. *Army, U.* Publication TM 5-315: Firefighting and Rescue Procedures in Theaters of Operation / U. Army. — US Army Corps of Engineers Internet Publishing Group., 05/1971. — Chap. Chapter 3: characteristics, chemistry, and physics of fire.
40. *Nielsen, T. E.* Modeling, animation, and visualization of fire : Master's thesis / Nielsen T. E. — University of Copenhagen, Denmark, 04/1999.
41. Детонация. — URL: <https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%82%D0%BE%D0%BD%D0%BC%D0%BD%D0%BD%D0%8F> (дата обр. 28.04.2020).
42. *Foster, N.* Modeling the motion of a hot, turbulent gas / N. Foster, D. N. Metaxas // SIGGRAPH '97. — 1997.
43. *Young, S. J.* Real Time Languages, Design and Development / S. J. Young // . — 1982.
44. *Perry, C. H.* Synthesizing Flames and their Spreading / C. H. Perry, R. W. Picard // Proceedings of the Fifth Eurographics Workshop on Animation and Simulation. — 1994. — P. 1–14.

45. *Woodhouse, F.* Particle systems: The theory / F. Woodhouse. — 2002.
46. *Harris, T.* How Fire Works / T. Harris. — URL: <https://science.howstuffworks.com/environmental/earth/geophysics/fire.htm> (visited on 05/01/2020).