**Основная** **теория**

Рендеринг в компьютерной графике означает процесс преобразования трехмерной сцены или объектов в двумерное изображение, которое будет отображаться на экране. Этот процесс включает в себя множество шагов, включая преобразование координат объектов, определение освещения, текстурирование, удаление невидимых поверхностей и применение различных эффектов.

Основные шаги рендеринга включают:

* Преобразование координат: Входные трехмерные координаты объектов преобразуются в экранные координаты, чтобы определить, как объекты будут отображаться на экране.
* Освещение: Определение освещения объектов с помощью различных моделей освещения, таких как фоновое освещение, точечные источники света и тени.
* Текстурирование: Наложение текстур на поверхности объектов, чтобы придать им более реалистичный вид.
* Удаление невидимых поверхностей: Определение, какие поверхности объектов находятся впереди, а какие скрыты другими поверхностями, чтобы избежать отображения невидимых объектов.
* Заливка и отрисовка: Заливка объектов цветом или текстурой и их отрисовка на экране.
* Применение эффектов: Добавление различных эффектов, таких как антиалиасинг, глубина размытия и других, для улучшения качества изображения.

Глубина цвета – число бит используемых для представления цвета каждого пиксиля.

Буфер кадров -

Рендеринг в один проход

Мощность рендеринга

Антиалиасинг

Буфер изображения

Спрайт

Полигон

Потоки вершин  
Шейдеры

Пиксельные шейдеры

Вершинные шейдеры

Текстурирование

Материал

Z-буфер

Триангуляция

Кубическая интерполяция

Интерполяция

Рейтрейсинг

Композитинг

Маппинг текстур

Шаг времени

Ортографическая проекция

Перспективная проекция

Глубина поля (Depth of Field)

Затенение

Проектирование камеры

Генеративное искусство (Generative Art): Искусство, созданное с помощью компьютерных программ и алгоритмов, которые генерируют уникальные и непредсказуемые результаты.

Клиппинг

Пост-обработка

Бамп-мэппинг

ресамплинге изображений

Скелетная анимация

Уровни детализации LOD

Загрузка текстур

Графический пайплайн

Графический процессор

**Дополнительная информация**

В играх часто используются различные методы оптимизации для предотвращения снижения FPS из-за не прогруженных объектов или других задержек. Некоторые из таких методов:

* Лоды (LOD - Level of Detail): Игры часто используют модели с разным уровнем детализации (LOD) для объектов. На дальних расстояниях используются более простые модели, чтобы уменьшить нагрузку на GPU и увеличить производительность.
* Потоковая загрузка ресурсов (Streaming): Вместо загрузки всех ресурсов в начале игры, некоторые игры используют потоковую загрузку для динамической загрузки ресурсов по мере необходимости. Это позволяет уменьшить время загрузки и улучшить производительность.
* Оптимизация отрисовки: Игры оптимизируют процесс отрисовки, чтобы минимизировать количество отрисовываемых объектов, скрытых объектов и повторное использование ресурсов.
* Асинхронная загрузка: Игры могут использовать асинхронную загрузку ресурсов, чтобы не блокировать основной поток выполнения и продолжать обновлять кадры даже во время загрузки.
* Оптимизация работы с памятью: Игры могут использовать различные техники для оптимизации работы с памятью, такие как компактное хранение данных, кэширование и другие методы для улучшения производительности.
* Предварительная компиляция шейдеров: Шейдеры (программы, выполняющиеся на GPU для отрисовки) могут быть предварительно скомпилированы для улучшения производительности.

**Функции OpenGL**

GLuint - это тип данных, определенный в OpenGL (или его библиотеках) для представления беззнакового целого числа, используемого как идентификатор объекта (например, текстуры, буфера вершин, шейдеры и т. д.).

RenderSceneCB – функция, которая обновляет кадр на экране при необходимости. Обычно в ней размещаются объекты, которые нужно отрисовать.

Функция InitializeGlutCallbacks() используется для связывания функции RenderSceneCB() с функцией отображения GLUT (glutDisplayFunc).

Функция отображения glutDisplayFunc в библиотеке GLUT используется для установки обратного вызова, который определяет, как будет отображаться содержимое окна при его обновлении.

Функция glutSwapBuffers() используется в контексте двойной буферизации (double buffering) в библиотеке GLUT (OpenGL Utility Toolkit) и позволяет переключить буферы, чтобы обновить содержимое окна. Когда используется двойная буферизация, вместо того, чтобы отображать результаты рисования прямо на экране, они временно сохраняются в неотображаемом "фоновом" буфере (заднем буфере). Когда рисование закончено и изображение готово, происходит переключение буферов: содержимое заднего буфера становится передним, и наоборот.

Функция glutMainLoop() является центральным элементом управления в библиотеке GLU. Она запускает основной цикл обработки событий и позволяет GLUT обрабатывать события окна, клавиатуры, мыши и другие пользовательские события.

glCreateProgram является функцией, предназначенной для создания программного объекта. В OpenGL программный объект представляет собой контейнер, который объединяет вершинный и фрагментный (или другие) шейдеры вместе. Шейдеры являются программами, выполняющими определенные вычисления на графическом процессоре, и они используются для рендеринга графики.