Колледж Автономной некоммерческой образовательной организации

высшего образования

«Научно-технологический университет «Сириус»

**Реферат**

**Технология LOD, ее история и современные аналоги**

Выполнил:

студент группы К0709-24/2

Перов Даниил Владимирович

Проверил: преподаватель

Яковлева Софья Вячеславовна

Сириус, 2025 г.

**Оглавление**

Введение

1 ТЕХНОЛОГИЯ LOD

2 ИСТОРИЯ ТЕХНОЛОГИИ

3 ПРИМЕНЕНИЕ ТЕХНОЛОГИИ LOD

4 СОВРЕМЕННЫЕ АНАЛОГИ

Современные аналоги LOD

# **Введение**

Современные технологии позволяют людям воспроизводить на экранах устройств с графическим процессором изображение, максимально приближенное к реальности. Один из побочных феноменов такой возможности – компьютерные игры (программы, служащие для организации игрового процесса), индустрия которых активно развивается с конца XX века.

Интерес представляют не только сами игры, но и процесс их создания: разработчики выбирают стилистику, движок, делают модели, рисуют текстуры для своих творений. Одним из важнейших аспектов в создании таких программ является оптимизация, ведь это – прямой ключ к коммерческой выгоде проекта.

Оптимизация – еще один многогранный процесс. Существует огромное количество методов, позволяющих увеличить производительность игры, и один из таких методов – LOD (Level of Detail).

**1 ТЕХНОЛОГИЯ LOD**

В компьютерной графике уровень детализации (LOD) относится к приемам представления 3D-модели. LOD может уменьшаться по мере удаления модели от зрителя или в соответствии с другими метриками, такими как важность объекта, скорость или положение относительно точки обзора. Методы LOD повышают эффективность рендеринга, снижая нагрузку на этапы графического конвейера, обычно вершинные преобразования. Снижение визуального качества модели часто остается незамеченным из-за незначительного влияния на внешний вид объекта при удалении или быстром движении.

Хотя чаще всего LOD применяется только для детализации геометрии, базовая концепция может быть обобщена. В последнее время методы LOD также включают управление шейдерами для контроля сложности пикселей. Одна из форм управления уровнем детализации уже много лет применяется к текстурным картам под названием mipmapping, также обеспечивая более высокое качество рендеринга.

Принято говорить, что «объект был LOD-ирован», когда объект упрощен как базовым алгоритмом LOD, так и 3D-художником, вручную создающим LOD-модели.

Хотя представленный выше алгоритм охватывает целый ряд методов управления уровнем детализации, в реальных приложениях обычно используются специализированные методы, учитывающие особенности отображаемой информации. В зависимости от требований ситуации используются два основных метода:

Первый метод, Discrete Levels of Detail (DLOD), предполагает создание нескольких дискретных версий исходной геометрии с пониженными уровнями геометрической детализации. Во время выполнения модели с полной детализацией заменяются на модели с пониженной детализацией по мере необходимости. Из-за дискретной природы уровней при замене одной модели на другую может возникнуть визуальный эффект. Это можно смягчить с помощью альфа-смешивания или морфинга между состояниями во время перехода.

Второй метод, Continuous Levels of Detail (CLOD), использует структуру, содержащую постоянно изменяющийся спектр геометрических деталей. Затем структуру можно прощупать, чтобы плавно выбрать подходящий уровень детализации, необходимый для конкретной ситуации. Существенным преимуществом этой техники является возможность локального изменения детализации; например, сторона крупного объекта, расположенного ближе к зрителю, может быть представлена с высокой детализацией, при этом одновременно снижается детализация его удаленной стороны.

**2 ИСТОРИЯ ТЕХНОЛОГИИ LOD**

Истоки всех алгоритмов LOD для трехмерной компьютерной графики восходят к статье Джеймса Кларка (James H. Clark) в октябрьском номере Communications of the ACM за 1976 год. В то время компьютеры были монолитными и редкими, а графикой занимались исследователи. Само аппаратное обеспечение было совершенно другим, как в архитектурном плане, так и в плане производительности. В связи с этим можно было заметить много различий с современными алгоритмами, но также и много общего.

Оригинальный алгоритм представлял собой гораздо более общий подход по сравнению с тем, что будет обсуждаться здесь. После представления некоторых доступных алгоритмов управления геометрией было заявлено, что наиболее плодотворные результаты были достигнуты за счет «...структурирования визуализируемого окружения», что позволило использовать более быстрые преобразования и операции обрезки.

Такое же структурирование окружения теперь предлагается в качестве способа управления различной детализацией, что позволяет избежать ненужных вычислений, обеспечивая при этом адекватное визуальное качество: например, додекаэдр выглядит как сфера с достаточно большого расстояния, и поэтому его можно использовать для моделирования, пока он рассматривается с этого или большего расстояния. Однако если его нужно рассмотреть более близко, он будет выглядеть как додекаэдр. Одно из решений этой проблемы - просто определить его с наибольшей детализацией, которая когда-либо потребуется. Однако в этом случае детализация может оказаться гораздо выше, чем требуется для представления на больших расстояниях, а в сложной среде с большим количеством подобных объектов будет слишком много многоугольников (или других геометрических примитивов), чтобы алгоритмы видимой поверхности могли эффективно с ними справиться.

Предлагаемый алгоритм представляет собой древовидную структуру данных, в дугах которой заложены преобразования и переходы к более детальным объектам. Таким образом, каждый узел кодирует объект, и в соответствии с быстрой эвристикой дерево спускается к листьям, которые обеспечивают более подробную детализацию каждого объекта. Когда лист достигнут, можно использовать другие методы, когда требуется более высокая детализация, например, рекурсивное подразделение Кэтмулла.

Однако важным моментом является то, что в сложном окружении количество информации о различных объектах варьируется в зависимости от доли поля зрения, занимаемой этими объектами.

Далее в статье рассматривается обрезка (не путать с выделением, хотя часто они похожи), различные соображения о графическом рабочем наборе и его влиянии на производительность, взаимодействие предлагаемого алгоритма с другими для повышения скорости рендеринга.

**3 ПРИМЕНЕНИЕ ТЕХНОЛОГИИ LOD**

LOD особенно полезен в 3D-видеоиграх. Разработчики видеоигр хотят предоставить игрокам большие миры, но их всегда ограничивают аппаратные средства, частота кадров и природа графики видеоигр в реальном времени. С появлением 3D-игр в 1990-х годах многие видеоигры просто не рендерили удаленные структуры или объекты. Прорисовывались только близкие объекты, а более удаленные части постепенно исчезали, по сути, создавая туман расстояния. Видеоигры, использующие LOD-рендеринг, позволяют избежать этого эффекта тумана и прорисовывать большие пространства. Среди ранних примеров использования LOD-рендеринга в 3D-видеоиграх можно назвать The Killing Cloud, Spyro the Dragon, Crash Bandicoot: Warped, Unreal Tournament и движок Serious Sam. В большинстве современных 3D-игр используется комбинация методов рендеринга LOD: различные модели для крупных структур и выделение расстояний для деталей окружения, таких как трава и деревья. Иногда этот эффект все же заметен, например, когда персонаж игрока летит над виртуальной местностью или использует снайперский прицел для наблюдения с большого расстояния. Особенно трава и листва будут казаться всплывающими при приближении, что также известно под названием foliage culling. LOD также может быть использован для рендеринга фрактальной местности в реальном времени.

LOD встречается в GIS и 3D-моделях городов как схожая концепция. Она показывает, насколько тщательно были отображены реальные объекты и насколько модель соответствует своему реальному аналогу. Помимо геометрической сложности, при определении LOD модели могут учитываться и другие метрики, такие как пространственно-семантическая связность, разрешение текстуры и атрибутов. Стандарт CityGML содержит одну из наиболее заметных классификаций LOD. Аналог «LOD-инга» в GIS называется генерализацией.

**4 СОВРЕМЕННЫЕ АНАЛОГИ LOD**

В настоящее время технология LOD все еще применяется и в комплексе с другими средствами позволяет кратно повышать производительность.

Nanite – это система виртуализированной геометрии движка Unreal Engine 5, которая использует новый внутренний формат сетки и технологию рендеринга для отображения деталей пиксельного масштаба и большого количества объектов. Она интеллектуально обрабатывает только те детали, которые могут быть восприняты, и не более того. Формат данных Nanite также отличается высокой степенью сжатия и поддерживает тонкую потоковую передачу с автоматическим уровнем детализации.

В большинстве случаев Nanite очень хорошо масштабируется в пределах разрешения экрана. Это происходит благодаря двум технологиям: тонкой детализации и устранению окклюзии. Как правило, это означает, что независимо от геометрической сложности исходных данных в сцене, количество треугольников, которые Nanite пытается нарисовать на экране, неизменно и пропорционально количеству пикселей.

Nanite следует принципу, что нет смысла рисовать больше треугольников, чем пикселей.

**Заключение**

Технология LOD (Level of Detail) играет важную роль в современной компьютерной графике и 3D-моделировании. Она позволяет оптимизировать производительность приложений, снижая нагрузку на вычислительные ресурсы за счет динамического изменения уровня детализации объектов. Это особенно актуально для игр, виртуальной и дополненной реальности, а также инженерных и архитектурных визуализаций.

Использование LOD обеспечивает баланс между качеством графики и производительностью, позволяя разрабатывать более сложные и масштабные виртуальные среды. С развитием технологий и увеличением вычислительных мощностей подходы к реализации LOD продолжают совершенствоваться, что делает данную технологию актуальной и востребованной в различных сферах компьютерной графики.

**Список литературы**

1 Level of Detail (LOD) - Unity Manual [Электронный ресурс] // Unity. — 2025. — URL: <https://docs.unity3d.com/ru/2017.4/Manual/LevelOfDetail.html> (дата обращения: 11.03.2025).

2 What is LOD (Level Of Detail) in 3D Modeling? [Электронный ресурс] // cgspectrum. — 2025. — URL: <https://www.cgspectrum.com/blog/what-is-level-of-detail-lod-3d-modeling> (дата обращения: 11.03.2025).

3 Nanite Virtualized Geometry [Электронный ресурс] // Epic Games. — 2025. — URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/nanite-virtualized-geometry-in-unreal-engine> (дата обращения: 11.03.2025).

4 Level Of Detail [Электронный ресурс] // Википедия. — 2025. — URL: <https://ru.wikipedia.org/wiki/Level_of_Detail> (дата обращения: 11.03.2025).