https://doi.org/10.25559/SITITO.022.202601.155-162
УДК 004.925.8:004.42.045

Оригинальная статья

# Анализ современных методов повышения производительности игровых приложений на базе Unreal Engine 5

**Д. И. Масталыгин**
ФГАОУ ВО «Московский политехнический университет», г. Москва, Российская Федерация
Адрес: 107023, Российская Федерация, г. Москва, ул. Большая Семёновская, д. 38
danya.masta@mail.ru

## Аннотация

Тотальная экспансия игрового движка Unreal Engine 5, который стал в 2024 году технологическим «фундаментом» для подавляющего большинства новых проектов, породила в индустрии острый инженерно-экономический парадокс. С одной стороны, внедрение систем виртуализированной геометрии (Nanite) и глобального освещения (Lumen) декларирует отказ от рутинных методов оптимизации, обещая «бесшовный» творческий процесс. С другой, эмпирические данные указывают на критический разрыв между вычислительной стоимостью этих технологий и реальными возможностями массового пользовательского оборудования, что ставит под угрозу коммерческую успешность продуктов. В данной статье предпринимается попытка разрешения этого противоречия через анализ архитектуры рендеринга UE5. Целью служит не просто перечисление настроек, а формирование комплексной методологии адаптации «тяжелого» некстген-контента под ограничения видеокарт среднего ценового сегмента. Автор фокусируется на поиске равновесия, при котором кинематографическое качество изображения сохраняется без драматического падения кадровой частоты. В ходе работы вскрывается неочевидная проблематика: отмечается, что слепое следование маркетинговым обещаниям «автоматической оптимизации» ведет к появлению «узких мест» как на стороне GPU, так и CPU. Основной авторский вклад проявляется в обосновании неизбежности перехода к гибридным моделям рендеринга, в которых инновационные инструменты сочетаются с классическими подходами, а также в переоценке роли апскейлинга. В частности, аргументировано, что технологии временного суперразрешения (TSR) де-факто перестали быть опциональным «костылем»; они превратились в неотъемлемый элемент графического конвейера. Сформулированные в статье выводы и прикладные рекомендации представляют существенную ценность для технических директоров, программистов графики. Эти данные позволяют принимать взвешенные архитектурные решения еще на этапе пре-продакшена, смягчая риски возникновения «технологического долга» к моменту релиза.

**Ключевые слова:** апскейлинг, виртуальная геометрия, глобальное освещение, игровая индустрия, оптимизация производительности, профилирование, рендеринг в реальном времени, Lumen, Nanite, Temporal Super Resolution, Unreal Engine 5

**Конфликт интересов:** автор заявляет об отсутствии конфликта интересов.

**Для цитирования:** Масталыгин Д. И. Анализ современных методов повышения производительности игровых приложений на базе Unreal Engine 5 // Современные информационные технологии и ИТ-образование. 2026. Т. 22, № 1. С. 155-162. https://doi.org/10.25559/SITITO.022.202601.155-162

Original article

# Analysis of Contemporary Methods for Improving the Performance of Game Applications Based on Unreal Engine 5

**D. I. Mastalygin**
Moscow Polytechnic University, Moscow, Russian Federation
Address: 38 Bolshaya Semyonovskaya St., Moscow 107023, Russian Federation
danya.masta@mail.ru

## Abstract

The total expansion of the Unreal Engine 5 game engine, which by 2024 had become the technological backbone for the vast majority of new projects, has generated a sharp engineering and economic paradox within the industry. On the one hand, the introduction of virtualized geometry systems (Nanite) and global illumination (Lumen) proclaims a departure from traditional optimization techniques, promising a "seamless" creative workflow. On the other hand, empirical data reveal a critical mismatch between the computational cost of these technologies and the actual capabilities of mass-market user hardware, thereby threatening the commercial viability of game products. This article seeks to resolve this contradiction through an in-depth analysis of Unreal Engine 5's rendering architecture. The objective is not merely to enumerate configuration parameters, but to develop a comprehensive methodology for adapting resource-intensive next-generation content to the constraints of mid-range graphics hardware. The author focuses on identifying an equilibrium in which cinematic visual fidelity can be preserved without a dramatic loss in frame rate. The study exposes a non-obvious issue: uncritical reliance on marketing narratives of "automatic optimization" leads to the emergence of performance bottlenecks on both GPU and CPU levels. The principal contribution of the article lies in substantiating the inevitability of a transition toward hybrid rendering models, in which innovative tools are combined with classical optimization approaches, as well as in re-evaluating the role of upscaling technologies. In particular, it is argued that temporal super-resolution (TSR) technologies have de facto ceased to be an optional workaround and have become an integral component of the modern graphics pipeline. The conclusions and applied recommendations formulated in the article are of substantial value to technical directors and graphics programmers, enabling informed architectural decisions at the pre-production stage and mitigating the risk of accumulating "technological debt" by the time of release.

**Keywords:** upscaling, virtualized geometry, global illumination, game industry, performance optimization, profiling, real-time rendering, Lumen, Nanite, Temporal Super Resolution, Unreal Engine 5

**Conflict of interests:** The author declares no conflict of interests.

**For citation:** Mastalygin D.I. Analysis of Contemporary Methods for Improving the Performance of Game Applications Based on Unreal Engine 5. *Modern Information Technologies and IT-Education.* 2026;22(1):155-162. https://doi.org/10.25559/SITITO.022.202601.155-162

## Введение

Современная индустрия разработки интерактивных развлечений переживает период фундаментальных технологических преобразований. Они вызваны, главным образом, сменой парадигм рендеринга и обработки графического контента. Если ранее повышение визуального качества линейно зависело от усложнения полигональной сетки и предварительного запекания освещения, то сегодня наблюдается переход к динамическим системам, которые способны обрабатывать кинематографическую геометрию в реальном времени.

Центральное место в рассматриваемом процессе, безусловно, занимает игровой движок Unreal Engine 5 (UE5). Его архитектура базируется на революционных технологиях Nanite и Lumen. Эти инструменты де-факто установили новый отраслевой стандарт. Между тем, они же создали беспрецедентные вызовы в области оптимизации производительности.

Значимость обращения к теме продиктована стремительным ростом популярности данной технологической платформы. Согласно статистическим данным [1], к 2024 году доля проектов на базе Unreal Engine 5 среди релизов на платформе Steam достигла внушительных 72%. Это указывает на массовую миграцию разработчиков на этот инструментарий. Такие флагманские проекты, как Black Myth: Wukong и S.T.A.L.K.E.R. 2, наглядно продемонстрировали визуальный потенциал движка. Но одновременно они вскрыли острую проблематику аппаратных ограничений. Для широкого круга пользователей, не обладающих топовыми графическими ускорителями, вопрос производительности становится критическим барьером.

Обозначенная проблема усугубляется тем, что базовые настройки UE5, которые ориентированы на демонстрацию визуальных возможностей, зачастую избыточны для реальных игровых сценариев. Внедрение технологий виртуализированной геометрии и глобального освещения в реальном времени требует от разработчиков не стандартного механического применения настроек, а глубокого понимания архитектуры рендеринга. Вследствие этого возникает объективная потребность в систематизации методов оптимизации, которые помогли бы сохранить визуальную идентичность проектов нового поколения, обеспечив при этом приемлемую частоту кадров на массовом оборудовании.

В рамках статьи будут рассмотрены эмпирические данные о нагрузке ключевых подсистем движка, сопоставлена эффективность различных методов масштабирования разрешения, сформулированы научно обоснованные рекомендации по адаптации контента. Особое внимание будет уделено балансу между использованием инновационных функций (Lumen, Nanite) и классических методов оптимизации, что составляет новизну исследования в контексте текущих реалий индустрии.

Совокупность изученных при подготовке статьи источников отражает многослойный характер исследований производительности игровых приложений на базе Unreal Engine 5, объединяя технологический, сравнительно-аналитический, прикладной подходы.

Так, рыночные и отраслевые обзоры [1] и аналитические материалы о метриках производительности [2] формируют контекст распространённости движка и задают систему ключевых показателей (FPS, frame time, latency), на основе которых оцениваются эффекты оптимизации. Технологически ориентированные исследования K. Chen, Ю.И. Таганова, А.С. Ягодкина, С.А. Солнышкина, А.А. Трунова, И.П. Проскурина, С.Ш. Джапарова, Т.С. Сулейманова акцентируют архитектурные особенности Unreal Engine 5 и эволюцию его инструментов по сравнению с предыдущими версиями; подчёркивается роль Lumen, Nanite, новых пайплайнов рендеринга [3-7]. Экспериментально-сравнительный подход реализован в работах M. Bao и соавт., А.А. Разживина и др., где производительность UE5 сопоставляется с альтернативными движками и версиями. Выявляется компромисс между визуальным качеством и вычислительными затратами [8, 9]. Узкоспециализированные публикации и официальная документация Epic Games, а также обзоры T. Looman, фокусируются на конкретных методах оптимизации — Temporal Super Resolution, настройке рендеринга, трассировке лучей и управлении ресурсами GPU [10-22]. Ряд работ (A. Aryanto, H. Xing, J. Verón) расширяет проблематику за счёт применения ИИ, *meta-learning,* анализа энергопотребления. Однако эти исследования пока слабо интегрированы в общий дискурс оптимизации игровых приложений [23-25].

Пробелом в литературе остается недостаток комплексных эмпирических исследований, где оценивается совокупный эффект нескольких методов оптимизации в реальных игровых проектах, а также противоречие между стремлением к фотореализму и требованиями к стабильной производительности на массовом оборудовании.

При написании статьи применено сочетание системного обзора литературы, сравнительного анализа, экспериментального тестирования производительности и профилирования. Это позволяет обоснованно оценить результативность современных методов оптимизации Unreal Engine 5 в прикладном контексте.

## Материалы и методы

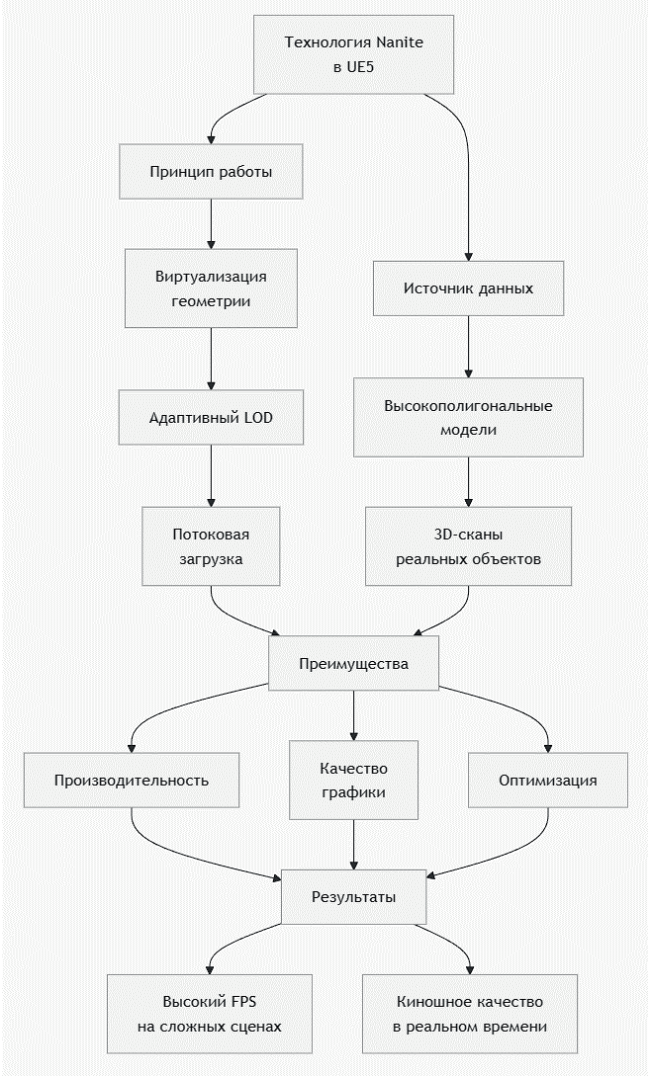
Unreal Engine – игровой движок, разрабатываемый и поддерживаемый компанией Epic Games. Первой игрой на этом движке был шутер от первого лица Unreal, выпущенный в 1998 году. Хотя движок первоначально был предназначен для разработки шутеров от первого лица, его последующие версии успешно применялись в играх самых различных жанров, в том числе, стелс-играх, файтингах, массовых многопользовательских ролевых онлайн-играх и многих других.

В прошлом движок распространялся на условиях оплаты ежемесячной подписки; с 2015 года Unreal Engine бесплатен, но разработчики использующих его приложений обязаны перечислять 5 % роялти от общемирового дохода с некоторыми условиями.

Компьютерная игра – программа, служащая для организации игрового процесса (геймплея), связи с партнёрами по игре, или сама выступающая в качестве партнёра. В настоящее время, в ряде случаев, вместо термина «компьютерная игра» может использоваться видеоигра, то есть, данные термины могут употребляться как синонимы и быть взаимозаменяемыми. В компьютерных играх, как правило, игровая ситуация воспроизводится на экране дисплея или обычного телевизора (в этом случае компьютерные игры одновременно являются и видеоиграми), но в то же время она может быть звуковой, телетайповой и другой.

Технология Nanite представляет собой систему виртуализированной геометрии, которая коренным образом меняет подход к

управлению детализацией (LOD) (рис. 1). В отличие от традиционных дискретных уровней, в Nanite используется кластерный рендеринг, автоматически подбирая детализацию на основе экранного разрешения. Сравнительные тесты, проведенные на сценах с высокой геометрической плотностью (модели по 3 млн полигонов), показывают, что Nanite превосходит классические системы LOD (например, в Unity) при визуализации объектов на средних и дальних дистанциях. Впрочем, следует отметить, что гибридное использование Nanite-геометрии с устаревшими методами обработки *(non-Nanite meshes)* может приводить к парадоксальному росту нагрузки на центральный процессор (из-за конфликта путей рендеринга) [3], [8], [13], [16].




Р и с. 1. Функционирование технологии Nanite в Unreal Engine 5
F i g. 1. Functioning of Nanite technology in Unreal Engine 5
(compiled by the author)

*Источник:* здесь и далее в статье все таблицы и рисунки составлены автором.
*Source:* Hereinafter in this article all tables and figures were made by the author.

Система глобального освещения Lumen, в свою очередь, является основным потребителем ресурсов GPU. Она функционирует в двух режимах: программной трассировки *(Software Ray Tracing)* и аппаратной *(Hardware Ray Tracing)* (рис. 2). Эмпирические замеры показывают, что на видеокартах среднего ценового сегмента одновременная активация Lumen и Nanite снижает производительность до критических 20-30 FPS. Это делает игровой процесс некомфортным. Версия движка 5.6 привнесла ряд низкоуровневых оптимизаций аппаратной трассировки. Вместе с тем, базовая «стоимость» кадра при включенном Lumen остается высокой, требуя от разработчиков гибкого подхода к настройке качества освещения [2], [11].

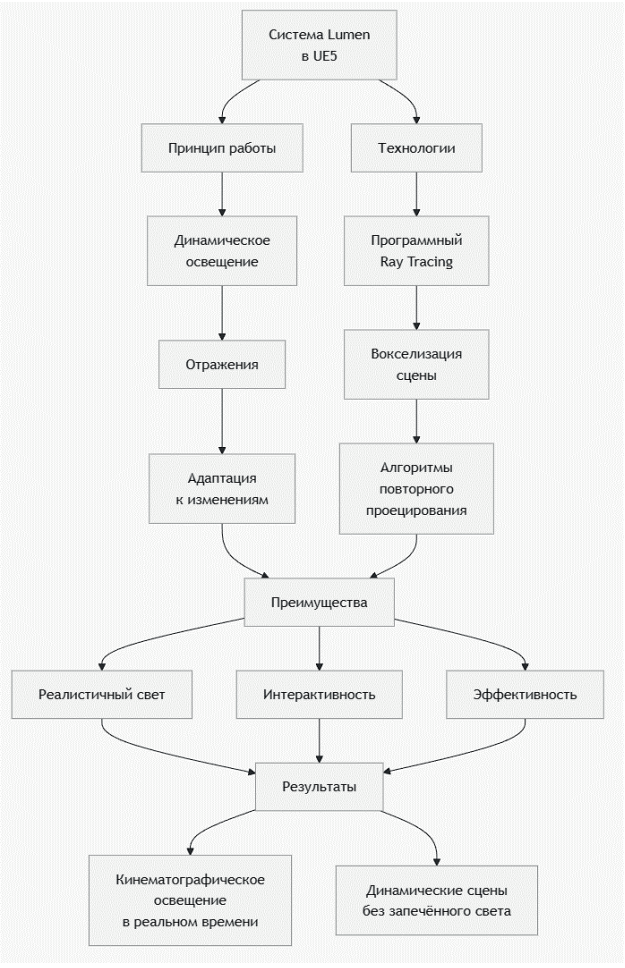
Для понимания масштаба проблемы целесообразно обратиться к конкретным данным. Тестирование промышленных демо-сцен (в частности, Industrial Factory) помогает обнаружить прямую зависимость между объемом видеопамяти (VRAM) и стабильностью работы. На высоких настройках потребление памяти приближается к 8 ГБ. Это является пределом для многих массовых видеокарт. Помимо этого, наблюдается регрессия производительности CPU в пятой версии движка по сравнению с четвертой, которая вызвана усложнением основного потока рендеринга [7], [12], [14], [17].

В таблице 1 представлены сводные данные относительно производительности, полученные в ходе независимых тестов на различных аппаратных платформах. Эти сведения наглядно иллюстрируют разрыв между «элитным» и «массовым» сегментами оборудования.

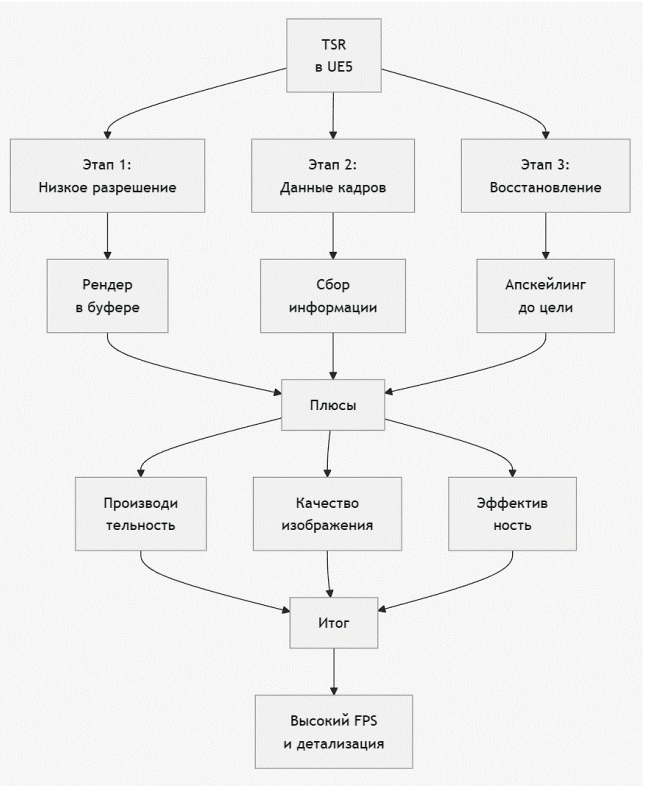
Информация, представленная в таблице 1, позволяет заключить, что для массового сегмента (GTX 1650, интегрированные решения) использование полных возможностей UE5 без дополнительных мер оптимизации невозможно.

Эффективная оптимизация нереальна без точной диагностики «узких мест». Unreal Engine 5 предоставляет мощный инструмент Unreal Insights (отдельная программа, которая по мере хода игры собирает различную полезную информацию и структурирует её), а также консольные команды ProfileGpu и Stat Unit. Использование последних дает возможность мгновенно определить, какой компонент системы (Game Thread, Draw Thread или GPU) является ограничивающим фактором. Версия 5.6 существенно расширила функционал GPU Profiler. Она позволяет детализировать затраты времени на проходы Lumen и генерацию виртуальных теневых карт (VSM).

Одним из наиболее действенных методов повышения производительности служит технология временного суперразрешения *(Temporal Super Resolution — TSR).* Она помогает рендерить изображение во внутреннем буфере с пониженным разрешением, а затем восстанавливать его до целевого разрешения экрана с задействованием данных из предыдущих кадров (рис. 3).




Р и с. 2. Функционирование системы глобального освещения Lumen в Unreal Engine 5
F i g. 2. Functioning of the Lumen global lighting system in Unreal Engine 5




Р и с. 3. Функционирование технологии временного суперразрешения в Unreal Engine 5
F i g. 3. Functioning of time super resolution technology in Unreal Engine 5

Т а б л и ц а 1. **Сравнительный анализ производительности UE5 на различных аппаратных конфигурациях (составлено на основе [1-3], [8], [11])**
T a b l e 1. **Comparative analysis of UE5 performance on various hardware configurations (compiled on the basis of [1-3], [8], [11])**

| Аппаратная платформа (GPU) | Режим настройки графики | Средняя частота кадров (FPS) | Время кадра (мс) | Примечание |
|---|---|---|---|---|
| NVIDIA RTX 4080 | Scalability: High / Epic, Lumen вкл. | ~115-120 | 8.3-8.5 | Стабильная работа в сложных сценах |
| NVIDIA RTX 3080 | Scalability: High, Lumen вкл. | ~100-110 | 9.0-10.0 | Оптимальный баланс для AAA-проектов |
| NVIDIA GTX 1650 Super | Scalability: Epic, Lumen + Nanite | 20-30 | 33.0-50.0 | Требуется агрессивный апскейлинг |
| Apple M4 (Mac Mini) | Scalability: Epic, Lumen + Nanite | ~20-21 | 47.0-48.0 | Нативное разрешение, неиграбельно |
| Apple M4 (Mac Mini) | Scalability: Epic, Lumen выкл. | ~29-30 | 34.0 | Отключение Lumen дает прирост ~40% |

Таблица 2. **Эффективность оптимизационных методов в среде Unreal Engine 5 (составлено на основе [4-7], [9], [12-19])**
Table 2. **Effectiveness of optimization methods in the Unreal Engine 5 environment (compiled on the basis of [4-7], [9], [12-19])**

| Метод оптимизации | Параметр настройки | Ожидаемый прирост производительности | Влияние на визуальное качество |
|---|---|---|---|
| Temporal Super Resolution | Screen Percentage: 65-75% | Высокий (30-50%) | Минимальное (артефакты в движении) |
| Упрощение Lumen | Global Illumination Quality: High → Med | Средний (15-20%) | Заметное (упрощение отражений и затенения) |
| Виртуальные тени (VSM) | Shadow Map Resolution: 2048 → 1024 | Умеренный (10-15%) | Низкое (смягчение краев теней) |
| Nanite Overdraw | Агрессивный culling невидимых мешей | Ситуативный (до 25%) | Отсутствует (при правильной настройке) |
| World Partition | Настройка дальности загрузки ячеек | Критический для RAM | Отсутствует (влияет на дальность прорисовки) |

Исследования показывают, что снижение параметра Screen Percentage до 50% в сочетании с TSR позволяет сократить время обработки кадра GPU почти вдвое – с ~0.79 мс до ~0.43 мс. При этом визуальное качество в динамике часто превосходит нативное 1080p с традиционным сглаживанием TAA[1] [23-25].

В таблице 2 систематизированы основные методы оптимизации и их прогнозируемое влияние на производительность.

Как видно из таблицы 2, наиболее выгодным соотношением «цена / качество» обладает грамотная настройка разрешения рендеринга (TSR). Отключение либо сильное снижение качества Lumen дает прирост, но ценой существенной деградации визуального стиля. А это не всегда приемлемо для художественных задач.

## Обсуждение и заключение

На основании проведенного анализа можно сформулировать ряд практических рекомендаций.

Во-первых, разработчикам необходимо внедрять системы адаптивного качества, динамически изменяющие разрешение рендеринга в зависимости от текущей нагрузки на GPU.

Во-вторых, в целях оптимизации проектов с открытым миром очень важно применение World Partition для управления памятью.

В-третьих, целесообразно придерживаться гибридного подхода – использовать Nanite для сложной статики, но сохранять классические LOD для динамических объектов и растительности, где выигрыш от виртуализации неочевиден.

В заключение целесообразно подчеркнуть, что переход индустрии на Unreal Engine 5 является необратимым процессом. Он, по-видимому, определит вектор развития компьютерной графики на ближайшее десятилетие. С помощью проведенного исследования подтверждается, что, несмотря на высокие системные требования, движок предоставляет достаточный инструментарий для масштабирования производительности. Технологии Nanite и Lumen, безусловно, являются прорывом. Между тем, их эффективное использование требует отказа от подхода «грубой силы» в пользу интеллектуальной оптимизации.

Статистика рынка, демонстрирующая доминирование UE5 (отмеченные ранее в статье 72% проектов в Steam в 2024 году), говорит о том, что разработчики готовы мириться со сложностями оптимизации ради предоставляемого инструментария и качества картинки. Вместе с тем, успех конечного продукта все чаще зависит не столько от художественного замысла, сколько от технической компетенции команды в вопросах профилирования и настройки рендеринга. В частности, с помощью эмпирических данных подтверждается, что технологии апскейлинга (к примеру, TSR) становятся не рядовой опцией, а обязательным элементом графического конвейера, помогающим преодолеть физические ограничения современного кремния.

Уместно предположить, что в ближайшем будущем фокус сместится с ручной настройки параметров на внедрение алгоритмов машинного обучения, которые способны автоматически балансировать настройки графики в реальном времени. Тем не менее, на текущем этапе глубокое понимание принципов работы Lumen, Nanite, виртуальной памяти остается фундаментальным навыком для создания конкурентоспособных игровых продуктов.

С авторской точки зрения, дальнейшие изыскания в этой области целесообразно сориентировать на изучение эффективности гибридных вычислительных моделей (CPU+GPU) для разгрузки основного потока рендеринга.

[1] Temporal Super Resolution. A high-level overview of the Anti-Aliasing options available in Unreal Engine [Электронный ресурс] // Epic Games Developers, 2026. https://dev.epicgames.com/documentation/en-us/unreal-engine/temporal-super-resolution-in-unreal-engine (дата обращения: 01.02.2026).

## References

[1] Ellul C., Hamilton N., Pieri A., et al. Exploring Data for Construction Digital Twins: Building Health and Safety and Progress Monitoring Twins Using the Unreal Gaming Engine. *Buildings.* 2024;14(7):e2216. https://doi.org/10.3390/buildings14072216
[2] Conde D., Balado J., Soilán M., et al. LiDAR Data Processing for Digitization of the Castro of Santa Trega and Integration in Unreal Engine 5. *International Journal of Architectural Heritage*. 2023;19(1):131-151. https://doi.org/10.1080/15583058.2023.2271877
[3] Chen K. Integrated application and development of Unreal Engine technology in gaming culture. *Lecture Notes in Education Psychology and Public Media.* 2025;106:95-101. https://doi.org/10.54254/2753-7048/2025.CB25229
[4] Taganov Yu.I. Comparison of XR development tools based on Unity and Unreal Engine. *Proceedings of the Seminar on Geometry and Mathematical Modeling.* 2023;9:81-83. (In Russ., abstract in Eng.) EDN: MQOOCU
[5] Yagodkin A.S., Solnyshkin S.A. Specific features of game development using Unreal Engine 5. In: Problematic Issues of Modeling Systems and Processes: Proceedings of the All-Russian Scientific and Practical Conference. Voronezh; 2024. p. 431-436. (In Russ., abstract in Eng.) https://doi.org/10.58168/CISMP2024_431-436
[6] Trunov A.A., Proskurin I.P. Modern software tools for computer game development. *Information Technologies in Education.* 2023;(6):322-325. (In Russ., abstract in Eng.) EDN: MUNSMW
[7] Dzhaparov S.Sh., Suleymanov T.S. Comparative analysis of Unreal Engine 4 and 5: advantages and new features. *Information and Computer Technologies in Economics, Education and Social Sphere*. 2025;(4):115-121. (In Russ., abstract in Eng.) EDN: GNDRUS
[8] Bao M., Tao Z., Wang X., Liu J., Sun Q. Comparative performance analysis of rendering optimization methods in Unity Tuanjie Engine, Unity Global, and Unreal Engine. In: 2024 IEEE Smart World Congress (SWC). Nadi, Fiji: IEEE Press; 2024. p. 1627-1632. https://doi.org/10.1109/SWC62898.2024.00250
[9] Razzhivin A.A., Limanova N.I., Kozlov V.V. Comparative analysis of game development platforms. *Bulletin of Science and Practice.* 2023;9(7):250-252. (In Russ., abstract in Eng.) https://doi.org/10.33619/2414-2948/92/35
[10] Del Blanco García F.L., González Cruz A.J., Amengual Menéndez C., et al. A Unified Virtual Model for Real-Time Visualization and Diagnosis in Architectural Heritage Conservation. *Buildings.* 2024;14(11):e3396. https://doi.org/10.3390/buildings14113396
[11] Wang Y. The virtual face modeling method based on user facial recognition and Unreal Engine 5 MetaHuman Creator. *Applied and Computational Engineering*. 2023;27(1):6-10. https://doi.org/10.54254/2755-2721/27/20230094
[12] Aditama P.W., Anggara I., Alparizi J. Using real-time ray tracing in the action-adventure game ANGKARA: The Rise of Asura. *Sinkron.* 2024;8(3):1967-1975. https://doi.org/10.33395/sinkron.v8i3.13826
[13] Díaz-Alemán M.D., Amador-García E.M., Díaz-González E., et al. Nanite as a disruptive technology for the interactive visualization of cultural heritage 3D models: A case study. *Heritage.* 2023;6(8):5607-5618. https://doi.org/10.3390/heritage6080295
[14] Ellul C., Hamilton N., Pieri A., et al. Exploring data for construction digital twins: building health and safety and progress monitoring twins using the Unreal gaming engine. *Buildings.* 2024;14(7):e2216. https://doi.org/10.3390/buildings14072216
[15] Fedotova N., Procenko M., Baranowa I., et al. Research on calculation optimization methods used in computer game development. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie* Środowiska. 2023;13(3):37-42. https://doi.org/10.35784/iapgos.3828
[16] Karpov D.E. Lumen and Nanite in Unreal Engine 5: analysis of new global illumination and visualization technologies, their impact on the development process and graphical quality. *Current Research.* 2024;(21-1):80-84. (In Russ., abstract in Eng.) EDN: KIUFQI
[17] Kassenova A., Temir E., Aitim A. Unreal Engine 5 and the development of the gaming industry in Kazakhstan: current capabilities and future prospects. *Bulletin of the Kazakh Academy of Transport and Communications named after M. Tynyshpaev.* 2024;(3):361-370. (In Russ., abstract in Eng.) https://doi.org/10.52167/1609-1817-2024-132-3-361-370
[18] Nikolaev I.S., Sagaeva I.D. Application of Unreal Engine tools for mobile application development. *Trends in the Development of Science and Education.* 2024;(110-17):162-168. (In Russ., abstract in Eng.) https://doi.org/10.18411/trnio-06-2024-956
[19] Orlov D.A., Belyaev P.V. Implementation of a state-saving system in a game application based on Unreal Engine. *Software Engineering*. 2025;16(12):646-654. (In Russ., abstract in Eng.) https://doi.org/10.17587/prin.16.646-654
[20] Osipov A.V., Pevchev I.S. Building a Computer Game Development Process and Ways to Increase its Effectiveness. In: Management Systems, Information Technologies and Mathematical Modeling: Proceedings of the V All-Russian Scientific and Practical Conference with International Participation. Omsk: OmSTU; 2023. p. 457-461. (In Russ., abstract in Eng.) EDN: IOLBQV
[21] Popov V.L., Shekhtman L.I., Lapin A.N. Analytical review of functional tools of modern game engines in the context of interactive application development. *Information Technologies: Problems and Solutions*. 2025;(1):90-96. (In Russ., abstract in Eng.) EDN: IILQWZ
[22] Khomichuk N.V., Zori S.A. Optimizing rendering performance in game engines using OpenCL technology. *Informatics and Cybernetics*. 2024;(4):5-11. (In Russ., abstract in Eng.) EDN: XSNTYI
[23] Aryanto A., Marccel Janara Brata Cipta I.N. Optimization of Unreal Engine 5 performance using meta-learning techniques for real-time rendering and AI adaptation. *Jurnal Pendidikan dan Teknologi Indonesia*. 2024;4(4):129-136. https://doi.org/10.52436/1.jpti.444
[24] Xing H., Chai M., Song Y. Artificial intelligence pathfinding based on Unreal Engine 5 hexagonal grid maps. In: 2024 4th International Conference on Neural Networks, Information and Communication (NNICE). Guangzhou, China: IEEE Press; 2024. p. 1708-1711. https://doi.org/10.1109/NNICE61279.2024.10498463

[25] Verón J., Pérez C., Calero C., et al. A comparative analysis of energy consumption between visual scripting models and C++ in Unreal Engine: raising awareness of the importance of green MDD. In: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems (MODELS '24). New York, NY, USA: Association for Computing Machinery; 2024. p. 114-125. https://doi.org/10.1145/3640310.3674099

*Поступила 01.02.2026; одобрена после рецензирования 20.03.2026; принята к публикации 07.04.2026.*
*Submitted 01.02.2026; approved after reviewing 20.03.2026; accepted for publication 07.04.2026.*

Об авторе:

**Масталыгин Даниил Игоревич**, студент факультета информационных технологий, ФГАОУ ВО «Московский Политехнический университет» (107023, Российская Федерация, г. Москва, ул. Большая Семёновская, д. 38), **ORCID: https://orcid.org/0009-0005-0724-2103,** danya.masta@mail.ru

*Автор прочитал и одобрил окончательный вариант рукописи.*

About the author:

**Daniil I. Mastalygin**, student of the Faculty of Information Technology, Moscow Polytechnic University (38 Bolshaya Semyonovskaya St., Moscow 107023, Russian Federation), **ORCID: https://orcid.org/0009-0005-0724-2103**, danya.masta@mail.ru

*The author has read and approved the final manuscript.*