УДК 378:004
DOI: 10.25686/978-5-8158-2474-4-2025-154-163

Гибридный метод отображения теней в реальном времени для систем с несколькими видеокартами

Р. Р. Султанов, Г. С. Лях, М. К. Богданов, А. М. Суворов
Университет ИТМО, г. Санкт-Петербург, Россия

Аннотация. Карты теней – наиболее распространенный способ расчета теней в приложениях реального времени, таких как компьютерные игры. В данной работе рассматривается применимость использования алгоритмов разделения задач между графическими процессорами на гибридных системах с несколькими видеокартами для увеличения производительности системы в целом. В исследовании предложен и протестирован алгоритм, разделяющий графический конвейер между двумя графическими процессорами с вынесением расчета теней на дополнительную видеокарту, в то время как на основной выполняются другие расчеты с последующим копированием данных перед отрисовкой освещения сцены. Для реализации использовался API DirectX 12, позволяющий реализовывать кросс-адаптерное взаимодействие. В результате разработанный алгоритм позволяет эффективно задействовать все доступные аппаратные ресурсы системы, обеспечивая при этом высокое качество и точность получаемых теней. Предложенное решение легко встраивается в существующие графические конвейеры и способствует увеличению производительности гибридных графических систем без значительных дополнительных затрат.

***Ключевые слова*:** карты теней, каскадные карты теней, DirectX 12, Multi-GPU, компьютерная графика

Hybrid method for displaying shadows in real-time in systems with multiple GPUs

R. R. Sultanov, G. S. Lyakh, M. K. Bogdanov, A. M. Suvorov
ITMO University, Saint Petersburg, Russia

Abstract. Shadow maps are the most common method for calculating shadows in real-time applications such as video games. This work explores the applicability of task-splitting algorithms across multiple graphics processors in hybrid systems with several GPUs to improve overall system performance. The study proposes and evaluates an algorithm that distributes the graphics pipeline between two GPUs, offloading shadow computations to the secondary graphics card while the primary one handles other rendering tasks, followed by data copying before the final lighting pass. The implementation was based on the DirectX 12 API, which enables cross-adapter communication. As a result, the developed algorithm efficiently utilizes all available hardware resources while maintaining high quality and accuracy of the computed shadows. The proposed solution integrates seamlessly into existing rendering pipelines and contributes to the performance increase of hybrid graphics systems without significant additional costs.

***Keywords*:** Shadow Maps, Cascaded Shadow Maps, DirectX 12, Multi-GPU, computer graphics

Введение

Современные компьютерные игры и графические приложения предъявляют всё более высокие требования к производительности графических систем. Реалистичная визуализация сцены включает множество параллельных алгоритмов: глобальное освещение, эффекты постобработки, моделирование сложных материалов. Особое место среди этих задач занимают тени в реальном времени, поскольку они одновременно повышают визуальную достоверность сцены и являются одним из наиболее ресурсоёмких этапов рендеринга.

Традиционные методы мульти-GPU рендеринга, такие как Alternate Frame Rendering (AFR) и Split Frame Rendering (SFR), хорошо изучены и применялись в графических системах прошлого поколения. Однако их использование в потребительских условиях ограничено: AFR подвержен проблеме межкадровых зависимостей и высокой задержке ввода, а SFR требует значительных затрат на синхронизацию и передачу данных. Поэтому возникает необходимость в исследовании более гибких подходов, в частности – гибридного рендеринга, при котором различные этапы графического конвейера распределяются между несколькими GPU.

Цель данной работы – исследовать возможность вынесения построения карт теней на дополнительный графический процессор (в том числе интегрированный) и определить эффективность подобного распределения в условиях домашних систем с ограниченной пропускной способностью межсоединения (PCI Express).

В этой работе:

1) предложена и реализована схема гибридного рендеринга, при которой вычисление карт теней выполняется на вспомогательном GPU, а основной адаптер обрабатывает остальные стадии конвейера;

2) показана применимость данного подхода для конфигураций с интегрированным и дискретным GPU без использования специализированных высокоскоростных интерфейсов (NVLink, XGMI);

3) проведён анализ производительности и выявлены условия, при которых межадаптерный обмен не нивелирует выигрыш от вынесения расчёта теней.

Обзор

Для систем с несколькими GPU существуют три базовых класса распределения работы: межкадровое (AFR), экранно-пространственное (SFR) и функционально-задачное (гибридное, по стадиям конвейера). Такая классификация восходит к классическим работам по параллельному рендерингу [1,2] и сегодня реализуема средствами современных графических API. В ряде обзоров подчеркивается, что выбор схемы распределения вычислений определяется архитектурой соединений и характером решаемых задач [3,4].

1. Alternate Frame Rendering (AFR). Каждый адаптер рендерит свой кадр (например, чётные/нечётные). Преимущества: простая декомпозиция и низкая связность между исполнителями [5,6]. Недостатки: рост полной задержки ввода-вывода на один кадр, «микростаттер» [5] при неоднородности производительности адаптеров, необходимость полной репликации ресурсов (сцены и состояний) на каждом GPU [7], а также ухудшение эффективности при наличии межкадровых зависимостей (TAA, TSR, SSR, временные тени и накопительные фильтры), характерных для современного рендеринга. На практике теоретическая близкая к линейной масштабируемость достигается редко.

Режим AFR исторически поддерживался драйверными технологиями SLI/CrossFire и специализированными режимами вроде VR SLI [8–10], однако в современных драйверах поддержка постепенно сокращается [8,9]. Современные работы также отмечают низкую эффективность AFR в условиях широкого использования темпоральных алгоритмов [6,11].

2. Split Frame Rendering (SFR). Кадр делится на регионы экрана (или изображение собирается композицией частичных результатов), каждый регион/подкадр отрисовывается на своём GPU[12,13]. Данный подход активно исследовался начиная с 1990-х годов [2,13]. Преимущества: потенциально лучшая балансировка нагрузки по сцено-зависимой сложности регионов, пониженная задержка по сравнению с AFR. Недостатки: необходимость дублирования/распределения геометрии, сложная синхронизация при прозрачностях и пересечениях регионов, высокие требования к пропускной способности межсоединения при композиции итогового изображения.

Более поздние работы предложили оптимизации: сокращение числа передаваемых примитивов [14], ассоциативная композиция в CHOPIN [15], использование иерархических структур для композиции [16]. Однако все эти решения предъявляют высокие требования к пропускной способности межсоединения и памяти [17].

Современные системы масштабируют SFR до десятков плат за счёт параллельной композиции: CHOPIN[15], MCM-GPU [18], работы по распределенным фреймбуферам [12,19]. Они демонстрируют хорошую масштабируемость, но требуют либо специализированных межсоединений (NVLink, XGMI) [20,21], либо высокопроизводительных сетевых технологий (Infiniband, RDMA) [13].

3. Гибридный Multi-GPU рендеринг. Разные стадии конвейера закрепляются за разными GPU: один считает тени/геометрию/постобработку, второй – основной кадр, и т.д. [22] Преимущества: снижение нагрузки на «главный» GPU и высокое перекрытие по времени, особенно для стадий с низкими межкадровыми зависимостями и компактным выходом (например, shadow maps, Hi-Z, SSAO). Недостатки: необходимость явного управления обменом данными и синхронизацией, а выигрыш ограничен полосой пропускания межсоединения и затратами на копирование.

Метод особенно эффективен там, где $T_{stage@dGPU} \gg T_{copy} + T_{overheads}$ [23], то есть если длительность стадии на основном GPU значительно превышает сумму времени передачи данных и накладных расходов, то ее выгодно вынести на второй GPU. В таком случае копирование можно полностью скрыть под другими стадиями [23].

В отличие от CHOPIN и аналогичных решений [15,18], в данной работе рассматриваются сценарии домашних ПК, где:

- взаимодействие ограничено PCIe (Gen3/Gen4/Gen5) [24, 25];
- отсутствует NVLink/XGMI [20, 21];
- возможны гетерогенные пары GPU (например, NVIDIA + Intel или NVIDIA + AMD), что подчёркивается в ряде исследований по гетерогенным вычислениям [26, 27].

API-аспекты (Direct3D 12, Vulkan). В Direct3D 12 реализованы два сценария: linked-node для «связанных» устройств и explicit multi-adapter для разнородных (включая связки «дискретная + интегрированная»). Поддерживаются кросс-адаптерные ресурсы и heap'ы и межадаптерные копии; синхронизация – через межадаптерные fences. В Vulkan аналогичная функциональность предоставляется через device groups и внешнюю память/семафоры. Эти механизмы являются базой для гибридных схем с переносом отдельных стадий на второй GPU.

В потребительских ПК обмен между адаптерами чаще всего идёт по PCIe; его пиковая пропускная способность ограничивает размер и количество межадаптерных копий. В рабочих станциях и HPC доступны NVLink и XGMI/IF Link с существенно большими скоростями и более низкими задержками [18,19,26], однако такие конфигурации нехарактерны для «домашних» систем. Выбор стратегии обмена данными и частоты синхронизаций должен учитывать именно эти аппаратные границы.

Метод Shadow Maps широко используется для реалистичного отображения теней в трёхмерной графике. Он был впервые предложен в 1978 году [28] и с тех пор применяется как в офлайн-рендеринге, так и в реальном времени. Идея заключается в том, что сцена рендерится с позиции источника света в буфер глубины (shadow map), после чего основной кадр отрисовывается из камеры с сравнением глубины каждого пикселя со значением из shadow map. Этот метод позволяет быстро получать как жёсткие, так и мягкие тени, поддерживает любые источники света и не требует сложных геометрических вычислений (в отличие от stencil-теней) [29].

Однако классические Shadow Maps имеют ограниченное разрешение текстуры глубины, из-за чего при большом расстоянии или большом угле обзора тени сильно искажаются (возникает перспективный «алиасинг», см. рис. 1). При удалении камеры от затенённых объектов несколько пикселей экрана могут проецироваться на один пиксел shadow map, приводя к размытию и «ступенчатым» артефактам. Проблема перспективного алиасинга считается одной из самых серьёзных при наложении теней в реальном времени.

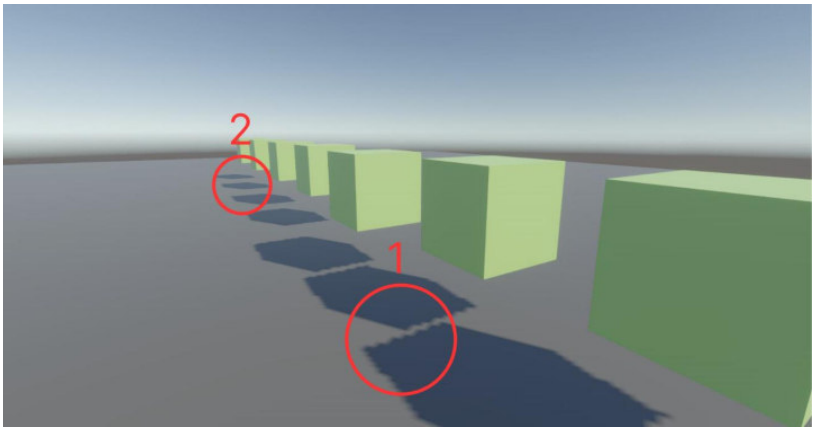

Рисунок 1. Демонстрация проблемы алиасинга

Perspective Shadow Maps (PSM) [30] использует перспективную проекцию от камеры, что повышает плотность теней на ближних расстояниях, но приводит к искажениям на дальних. Метод Light-space Perspective Shadow Maps (LiSPSM) [31] улучшает эту идею, учитывая геометрию сцены и направление света при формировании проекции, что обеспечивает лучшее распределение плотности теней. Также был предложен метод Trapezoidal Shadow Maps (TSM) [32], в котором форма shadow map деформируется в трапецию, приближенную к области обзора камеры, что уменьшает aliasing. Другие методы, такие как Sub-pixel Shadow Mapping [33], направлены на повышение качества теней за счёт

более точной фильтрации границ, а Irregular Z-Buffer [34] использует лучевое построение буфера глубины, полностью избавляясь от регулярной сетки shadow map. Однако оба эти метода требуют высокой вычислительной мощности и сложны в реализации.

Также для улучшения распределения разрешения по глубине сцены были предложены Parallel-Split Shadow Maps [35]и Cascaded Shadow Maps (CSM) [36]. В этих методах пирамида видимости камеры (фрустум) разбивается на несколько диапазонов (кассадов), каждый из которых покрывает свою часть сцены и имеет свою shadow map. Это позволяет выделить большую текстуру и более высокое разрешение теней для ближнего диапазона, где качество особенно важно, и меньше – для дальних объектов.

Современные графические движки также применяют более сложные техники: виртуальные карты теней (Virtual Shadow Maps) [36] благодаря концепции виртуальных текстур [37] позволяют создать карты теней очень высокого виртуального разрешения (например, 16384×16384 в UE5) с динамической загрузкой страниц.

Для данного исследования были выбраны методы Shadow Mapping и Cascaded Shadow Maps по нескольким причинам. Во-первых, они являются наиболее широко применяемыми в современной индустрии интерактивной графики, включая игровые движки и системы визуализации в реальном времени, что обеспечивает их практическую актуальность. Во-вторых, эти алгоритмы отличаются разумным балансом между качеством теней и вычислительной сложностью, что особенно важно при построении гибридных мульти-GPU систем. В отличие от более продвинутых методов, выбранные алгоритмы обладают более простой архитектурой и легче масштабируются в условиях нескольких видеокарт, не требуя сложной синхронизации или нестандартных этапов рендеринга. Кроме того, Virtual Shadow Maps, хотя и обеспечивают очень высокое качество, пока остаются слишком ресурсоёмкими и требуют тесной интеграции с кастомными рендерерами, что затрудняет их применение в гибридных системах без глубокого вмешательства в движок. Таким образом, выбор Shadow Mapping и CSM позволяет исследовать производительность и масштабируемость классических алгоритмов в условиях мультиадаптерной архитектуры, сохраняя применимость полученных решений в реальных проектах.

Постановка задачи

Цель данного исследования – реализовать и сравнить методы Shadow Mapping и Cascaded Shadow Maps в контексте гибридных графических систем с несколькими видеокартами. Основное внимание уделяется разработке и тестированию мульти-GPU версий этих алгоритмов, в которых расчёт карт теней переносится на дополнительный графический процессор, в то время как основной GPU выполняет остальные этапы рендеринга. Таким образом, задача состоит в том, чтобы:

1) реализовать оба алгоритма (Shadow Maps и Cascaded Shadow Maps) как в однопроцессорной конфигурации, так и в конфигурации с несколькими GPU;

2) измерить и сравнить производительность (время рендеринга кадра, FPS) в обоих режимах работы для разных сцен и наборов видеокарт;

3) оценить, даёт ли multi-GPU реализация преимущество в скорости по сравнению с классической однопроцессорной схемой;

4) провести анализ полученных данных, чтобы определить, в каких условиях использование второй видеокарты наиболее эффективно.

Теория

В данной работе расчёт карт теней полностью вынесен на вспомогательный адаптер (в т.ч. интегрированный GPU), в то время как основной GPU параллельно выполняет G-pass и эффекты, не зависящие от теней. По готовности shadow maps выполняется кросс-адаптерное копирование в локальную память дискретного GPU, после чего запускается проход освещения. Декомпозиция выбрана как слабо зависящая от данных предыдущего кадра и имеющая компактные выходные ресурсы, что позволяет перекрыть обмен с основными стадиями и минимизировать простои. Реализация опирается на механизмы Direct3D 12 explicit multi-adapter и кросс-адаптерные ресурсы [38]. Схема алгоритма представлена на диаграмме (рис. 2).

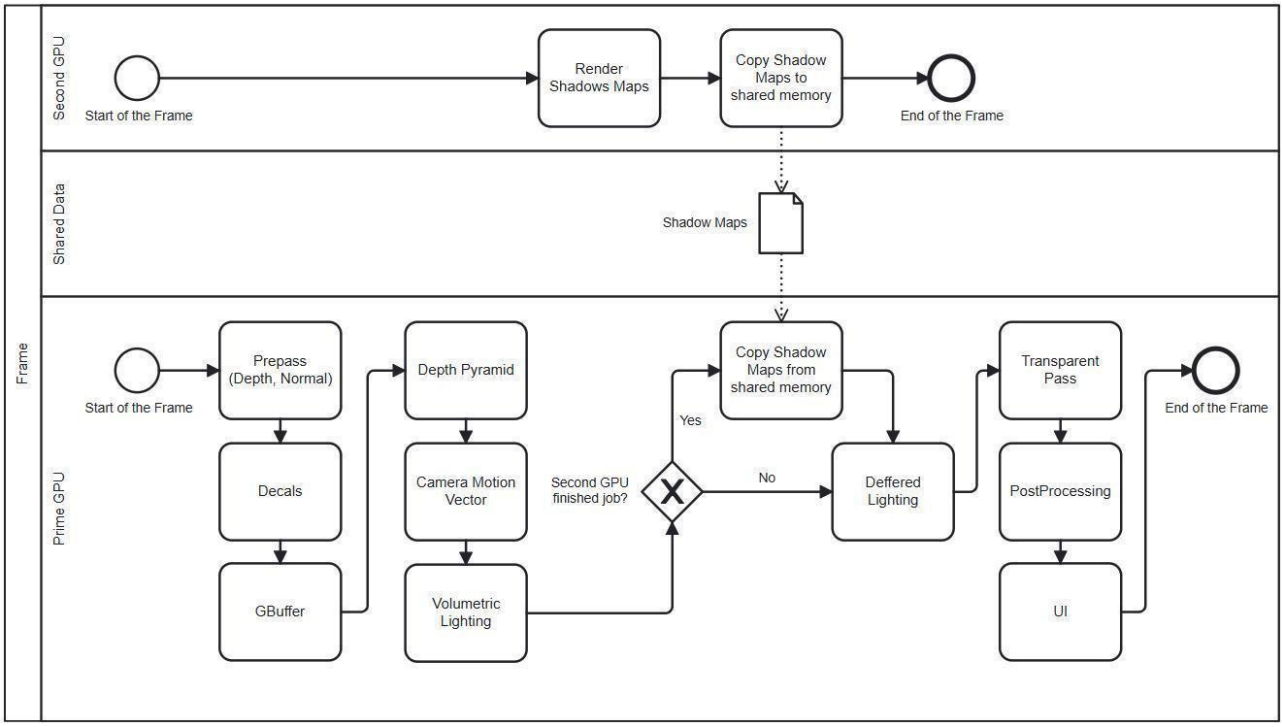

Рисунок 2. Диаграмма активности multi-GPU Shadow Mapping алгоритма

Классический алгоритм Shadow Maps базируется на рендеринге сцены из точки зрения источника света и сравнении глубин. Его этапы обычно таковы:

1. Создание shadow map: сцена рендерится с позиции источника света (обычно с использованием ортографической или перспективной проекции, в зависимости от типа света), при этом пишется лишь буфер глубины (рис. 3). Полученная карта глубины хранит расстояния до ближайших поверхностей относительно света.

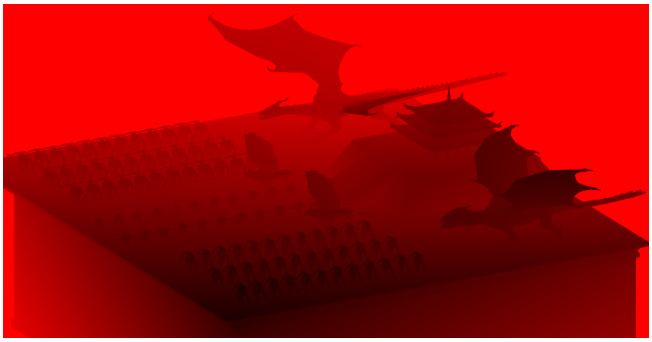

Рисунок 3. Пример shadow map

2. Отрисовка сцены камеры: затем выполняется обычная отрисовка сцены с точки зрения камеры, но при вычислении освещения для каждого фрагмента дополнительно определяется его координата в системе координат света и сравнивается его глубина с соответствующим значением из shadow map. Если глубина фрагмента больше, чем в shadow map, то этот фрагмент невидим для света и должен быть затенён.

Таким образом, Shadow Maps позволяют быстро определить теневые участки без сложных вычислений в пространстве изображения. Недостатком метода является ограниченность разрешения карты глубины. При необходимости покрыть большой объём сцены одна текстура shadow map не содержит достаточно пикселей, чтобы обеспечить плавные тени на больших расстояниях. Это приводит к артефактам перспективного искажения теней (blocky shadows, перспективный «aliasing»).

Метод каскадных теневых карт (CSM) решает эту проблему за счёт адаптивного распределения разрешения по расстоянию. Его суть заключается в разделении пирамиды видимости камеры на несколько диапазонов (сплитов) по глубине (рис. 4). Ближайшие к камере объекты попадают в первый

сплит и затеняются с помощью первой shadow map, второй диапазон – с помощью второй карты глубины и т.д. При этом ближним диапазонам камеры выделяется меньшая область пространства (но на той же текстуре shadow map), что повышает плотность пикселей теней там, где требуется больше деталей. Для дальнего диапазона используется отдельная карта теней, покрывающая больший объём, но его потеря в качестве менее заметна на удалении.

На рисунке 4 изображено типичное разбиение пирамиды видимости на три каскада при использовании CSM: ближний каскад (самый маленький объём) получает самую крупную часть разрешения, дальний – меньшую. Для каждого сплита вычисляются собственные матрицы вида и проекции света, что даёт возможность более точно подобрать область рендеринга shadow map под этот диапазон. Благодаря такому разбиению теней повышается качество вблизи камеры, а дальние области при необходимости рисуются более грубо.

Алгоритм CSM можно формализовать следующими шагами:

1. Разбить frustum камеры на n диапазонов по глубине.
2. При рендеринге shadow map: для каждого диапазона вычислить матрицы вида и проекции источника света и отрисовать сцену в отдельную текстуру глубины.
3. При финальной отрисовке сцены: для каждого фрагмента определить, к какому диапазону он принадлежит, и использовать соответствующую карту глубины для проверки тени.

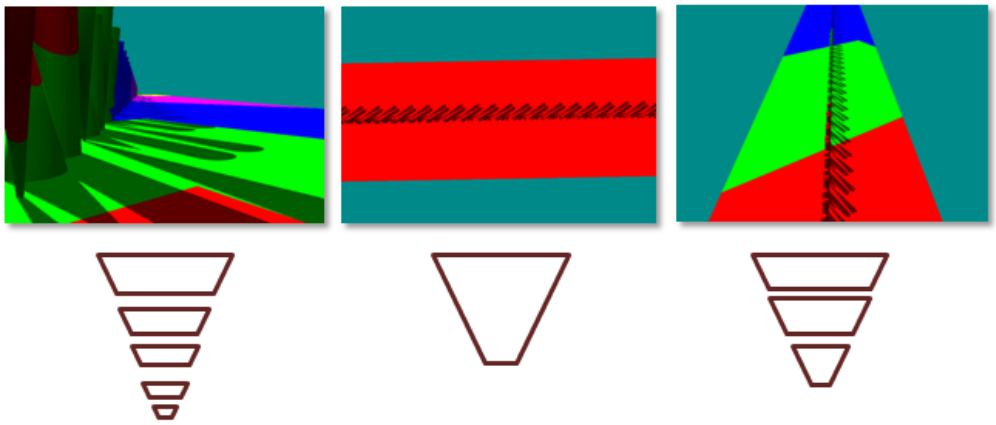

Рисунок 4. Разделение пространства на каскады, отмеченные разным цветом

Основной недостаток CSM – увеличение числа проходов рендеринга теневой карты. Если в обычном алгоритме достаточно одного рендера сцены от света, то в CSM таких проходов n (по числу каскадов). Это требует дополнительных вычислений и может снижать FPS. Тем не менее в задачах больших открытых миров и при дальних планах эта затрата оправдана за счёт видимого улучшения качества теней (уменьшения артефактов).

Результаты экспериментов

В рамках эксперимента были проведены замеры времени рендеринга и FPS на различных конфигурациях оборудования. Были протестированы две реализации алгоритмов Shadow Mapping и Cascaded Shadow Maps: стандартная (одна видеокарта) и гибридная (две видеокарты, с расчётом теней на вторичном GPU) при различных параметрах SSAA (SuperSample Anti-Aliasing).

1. Shadow Mapping

Наиболее существенное уменьшение времени $\varepsilon = (t_1 - t_2)/t_1$ на кадр, где t_1 – время рендеринга с использованием одной видеокарты, а t_2 – время рендеринга с использованием двух видеокарт, зафиксировано на системе с NVIDIA GeForce GTX 1650 и AMD Radeon(TM) Vega 8 Graphics – уменьшение составило 10 % (табл. 1). Среднее уменьшение для остальных систем также варьировалось от 5 до 10 %, демонстрируя положительный эффект от вынесения расчёта теней на вторую видеокарту. Для данных результатов использовалась сцена "Bistro" (2 829 238 треугольников)

На рисунке 5 представлены графики времени рендеринга при различных множителях SSAA. По результатам видно, что при увеличении разрешения (множители 2, 4 и 8) относительный выигрыш

от использования multi-GPU снижается. При множителе 8 выигрыш практически исчезает, что связано с тем, что основная нагрузка в этом случае смещается с расчёта теней на рендеринг основного кадра.

Таблица 1. **Время обработки кадра при использовании обычного метода Shadow Map на сцене "Bistro" (2 829 238 треугольников)**

Тестируемая система	SSAA x1			SSAA x3		
	t_1, мс	t_2, мс	ε	t_1, мс	t_2, мс	ε
NVIDIA GeForce GTX 980 + Intel(R) HD Graphics 630	9,26	8,85	4%	16,67	16,67	0%
NVIDIA GeForce GTX 1060 + Intel(R) UHD Graphics 630	12,20	10,99	10%	22,73	20,83	8%
NVIDIA GeForce GTX 1650 + AMD Radeon(TM) Vega 8 Graphics	15,38	14,08	8%	27,03	24,39	10%
NVIDIA GeForce GTX 1070 + Intel(R) HD Graphics 630	7,94	7,30	8%	14,29	13,33	7%
NVIDIA GeForce GTX 1080 + Intel(R) HD Graphics 630	6,10	5,46	10%	10,87	10,20	6%
NVIDIA GeForce GTX 1080 Ti + Intel(R) HD Graphics 630	4,98	4,52	9%	8,93	8,47	5%
NVIDIA GeForce RTX 2080 + Intel(R) UHD Graphics 630	4,55	4,88	3%	7,41	6,76	2%

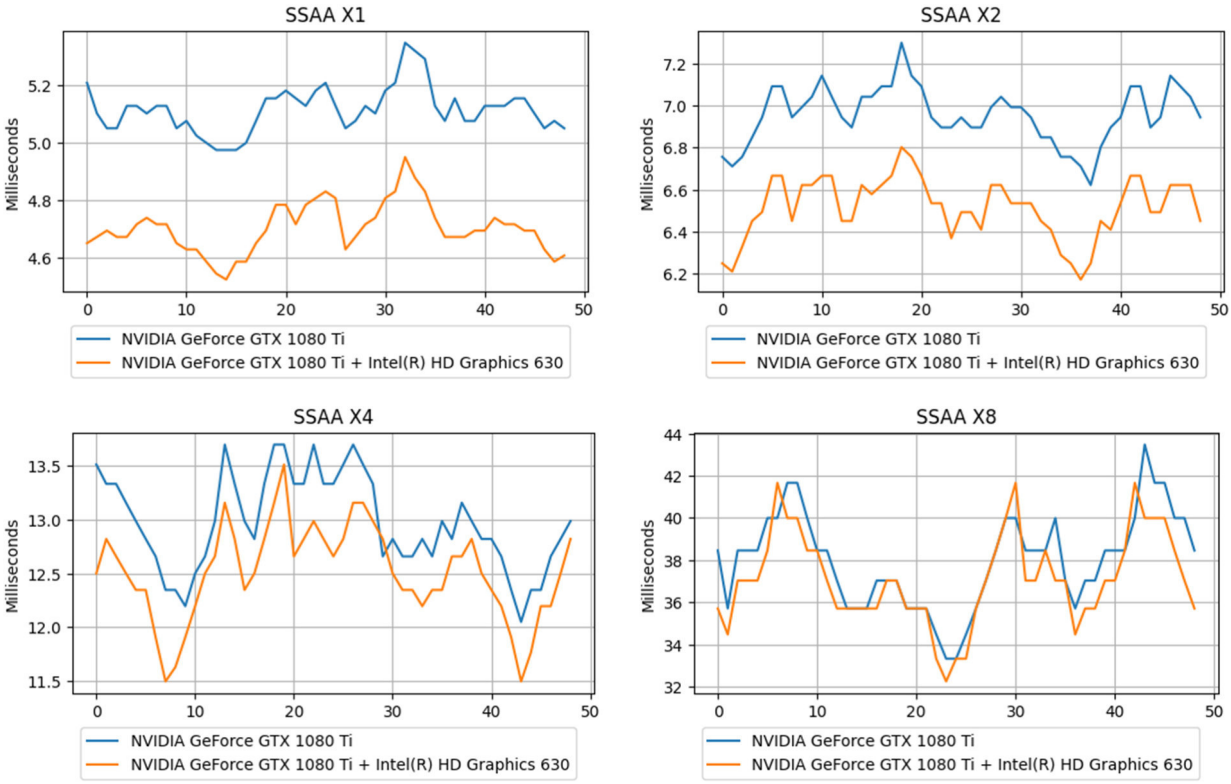

Рисунок 5. Графики времени рендеринга при различных множителях SSAA

1. Cascaded Shadow Maps

В трёх сценах разной сложности – "Pica Pica", "The Town on Capital Isle" и "Bistro" – фиксировалось заметное сокращение времени рендеринга в мульти-GPU варианте. Например, система с GTX 1650 Ti и UHD Graphics показала сокращение времени до 49 % на сцене "Bistro" (табл. 2–4). Система с RTX 3060 и Radeon Graphics показала от 20 до 44 % сокращение времени.

Видно, что мульти-GPU эффективно масштабирует именно те части рендеринга, которые можно выполнять независимо, – такие как расчёт карт теней. Однако общая эффективность зависит от архитектурной совместимости GPU, скорости передачи данных и объёма общей памяти.

Таблица 2. **Сокращение времени рендеринга для сцены "Pica Pica" (76 274 треугольников) при использовании CSM**

Тестируемая система	t_1, мс	t_2, мс	Сокращение времени рендера
NVIDIA GeForce GTX 1050 + Intel(R) UHD Graphics 630	2,73	1,97	28%
NVIDIA GeForce GTX 1650 Ti + Intel(R) UHD Graphics	1,41	1,08	23%
NVIDIA GeForce RTX 3060 + AMD Radeon(TM) Graphics	0,7	0,56	20%
NVIDIA GeForce RTX 4070 Laptop GPU + Intel(R) Iris(R) Xe Graphics	0,93	0,86	8%

Таблица 3. **Сокращение времени рендеринга для сцены "The Town on Capital Isle" (193 485 треугольников) при использовании CSM**

Тестируемая система	t_1, мс	t_2, мс	Сокращение времени рендера
NVIDIA GeForce GTX 1050 + Intel(R) UHD Graphics 630	3,61	2,66	26%
NVIDIA GeForce GTX 1650 Ti + Intel(R) UHD Graphics	2,16	1,5	31%
NVIDIA GeForce RTX 3060 + AMD Radeon(TM) Graphics	0,96	0,69	28%
NVIDIA GeForce RTX 4070 Laptop GPU + Intel(R) Iris(R) Xe Graphics	1,28	1,1	14%

Таблица 4. **Сокращение времени рендеринга для сцены "Bistro" (2 829 238 треугольников) при использовании CSM**

Тестируемая система	t_1, мс	t_2, мс	Сокращение времени рендера
NVIDIA GeForce GTX 1050 + Intel(R) UHD Graphics 630	55,74	42,09	24%
NVIDIA GeForce GTX 1650 Ti + Intel(R) UHD Graphics	91,52	46,42	49%
NVIDIA GeForce RTX 3060 + AMD Radeon(TM) Graphics	3,67	2,05	44%
NVIDIA GeForce RTX 4070 Laptop GPU + Intel(R) Iris(R) Xe Graphics	4,27	2,52	41%

Обсуждение результатов

Полученные данные демонстрируют, что мульти-GPU подход способен значительно улучшить производительность при расчёте теней. Особенно это заметно на сценах с высокой геометрической сложностью и большим количеством теневых проходов, где использование CSM даёт до 49 % выигрыша по времени.

При этом простой метод Shadow Mapping также выигрывает от распределения нагрузки, но его ускорение ограничивается примерно 10–16 % в зависимости от конкретной конфигурации.

Направления дальнейших исследований

Перспективным направлением развития предложенного подхода является более тонкое распределение задач между адаптерами. В частности, возможна дифференциация каскадов карт теней: ближние каскады, требующие минимальной задержки и высокой точности, могут вычисляться на основном дискретном GPU, в то время как дальние каскады, обладающие меньшими требованиями к качеству и скорости, могут быть вынесены на вспомогательный GPU. Подобное разделение позволит дополнительно снизить нагрузку на основной адаптер и более гибко использовать доступные аппаратные ресурсы.

Другим направлением может стать интеграция гибридного мульти-GPU подхода с другими алгоритмами, которые могут быть вынесены на вторичный GPU и при этом требуют доступа к данным о тенях. К таким задачам относятся, например, процедурная симуляция растительности (листвы, травы), расчёты водных поверхностей, а также дополнительные эффекты глобального освещения. Совмещение этих алгоритмов с вынесенным построением карт теней позволит повысить степень параллелизма и снизить задержку в критических стадиях конвейера.

Выводы

В работе была реализована мульти-GPU архитектура рендеринга теней, в которой расчёт shadow map переносится на вспомогательный графический адаптер. На её основе протестированы методы Shadow Mapping и Cascaded Shadow Maps в различных конфигурациях. Эксперименты показали, что такой подход позволяет достичь прироста производительности до 49 % на сценах с высокой детализацией.

Таким образом, построенный алгоритм эффективно использует доступные аппаратные ресурсы домашних ПК, позволяя разгрузить основной GPU. Разработка хорошо интегрируется в существующие графические конвейеры и применима в широком спектре визуализаторов реального времени, включая игровые движки и системы архитектурной визуализации.

Список литературы

1. A sorting classification of parallel rendering / Molnar S., et al. // IEEE Comput Graph Appl. 1994. Vol. 14, no. 4. P. 23–32.

2. Sort-first, sort-middle, sort-last parallel rendering / Humphreys G., Hanrahan P. // ACM SIGGRAPH Course Notes. 2002.

3. A Survey of General-Purpose Computation on Graphics Hardware / Owens J.D., et al. // Computer Graphics Forum. 2007. Vol. 26, no. 1. P. 80–113.

4. Pharr M., Fernando R., Humphreys G. GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation. Addison-Wesley, 2005.

5. Krawczyk G. Microstuttering in AFR-based multi-GPU rendering // Computer Graphics Forum. 2009. Vol. 28, no. 2. P. 367–374.

6. Haines E. Temporal reprojection and frame dependencies in rendering // ACM Queue. 2018. Vol. 16, no. 6. P. 40–61.

7. Interactive visibility ordering and transparency computations on GPUs / Govindaraju N.K., et al. // Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. 2005. P. 49–56.

8. NVIDIA. SLI Best Practices Guide. 2015.

9. VR SLI – Scalable Per-Eye Rendering for VR // NVIDIA Developer. URL: https://developer.nvidia.com/vrworks/graphics/vrsli (дата обращения: 01.09.2025).

10. SLI Best Practices and Rendering Modes // NVIDIA Developer. URL: https://docs.nvidia.com/gameworks/content/technologies/desktop/sli.htm (дата обращения: 01.09.2025).

11. Karis B. High-quality temporal supersampling // ACM SIGGRAPH Courses. 2014.

12. Moreland K., Angel E. The Distributed Framebuffer // Proceedings of the IEEE Symposium on Parallel Visualization and Graphics. 2001. P. 61–68.

13. Parallel volume rendering using binary-swap compositing / Ma K.-L., et al. // IEEE Comput Graph Appl. 1994. Vol. 14, no. 4. P. 59–68.

14. GPUpd: A fast and scalable multi-GPU architecture using cooperative projection and distribution / Kim Y., et al. // Proceedings of the Annual International Symposium on Microarchitecture, MICRO. IEEE Computer Society, 2017. Vol. Part F131207. P. 574–586.

15. Ren X., Lis M. CHOPIN: Scalable Graphics Rendering in Multi-GPU Systems via Parallel Image Composition // Proceedings - International Symposium on High-Performance Computer Architecture. IEEE Computer Society, 2021. Vol. 2021-February. P. 709–722.

16. Load balancing for sort-first parallel rendering / Samanta R., et al. // IEEE Trans Vis Comput Graph. 2001. Vol. 7, no. 4. P. 353–368.

17. Mueller C., Moorhead R., Moreland K. Evaluating bandwidth requirements for parallel rendering systems // IEEE Symposium on Parallel Visualization and Graphics. 2002. P. 103–110.

18. MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scaling / Jain A., et al. // Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA). 2017. P. 320–332.

19. Moreland K. Scalable Rendering on HPC Systems // Computer Graphics Forum. 2016. Vol. 35, no. 3. P. 485–494.

20. NVLink High-Speed GPU Interconnect // NVIDIA. URL: www.nvidia.com/en-us/data-center/nvlink (дата обращения: 01.09.2025).

21. Infinity Fabric Link // AMD. URL: www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/white-papers/overview-amd-epyc7003-series-processors-microarchitecture.pdf (дата обращения: 01.09.2025).

22. Computer Graphics: Principles and Practice / Foley J. et al. // 3rd изд. Addison-Wesley, 2014.

23. Aila T., Laine S. Understanding the efficiency of ray traversal on GPUs // Proceedings of the Conference on High Performance Graphics (HPG). 2009. P. 145–149.

24. PCI-SIG. PCI Express Base Specification 5.0. 2019.

25. Intel. PCI Express Gen5 Performance Overview. 2021.

26. Qilin: Exploiting parallelism on heterogeneous multiprocessors with adaptive mapping / Luk C.-K., et al. // Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). 2009. P. 45–58.

27. Diamos G.F., Yalamanchili S. Harmony: An execution model and runtime for heterogeneous many core systems // 2008 IEEE 15th International Symposium on High Performance Computer Architecture. 2008. P. 356–367.

28. Williams L. Casting curved shadows on curved surfaces. 1978.

29. Chapter 12. Omnidirectional Shadow Mapping // NVIDIA Developer. URL: https://developer.nvidia.com/gpugems/gpugems/part-ii-lighting-and-shadows/chapter-12-omnidirectional-shadow-mapping (дата обращения: 01.09.2025).

30. Stamminger M., Drettakis G. Perspective shadow maps // Proceedings of the 29th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, 2002. P. 557–562.

31. Wimmer M., Scherzer D., Purgathofer W. Light Space Perspective Shadow Maps. 2004.

32. Anti-aliasing and Continuity with Trapezoidal Shadow Maps / Jensen H.W., et al. // Eurographics Symposium on Rendering. 2004.

33. Sub-pixel shadow mapping / Lecocq P., et al. // Proceedings of the 18th meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. New York, NY, USA: ACM, 2014. P. 103–110.

34. Johnson G.S., Mark W.R., Burns C.A. The Irregular Z-Buffer and its Application to Shadow Mapping. 2004.

35. Parallel-split shadow maps on programmable GPUs. URL: https://developer.nvidia.com/gpugems/gpugems3/part-ii-light-and-shadows/chapter-10-parallel-split-shadow-maps-programmable-gpus (дата обращения: 01.09.2025).

36. Dimitrov Rouslan. Cascaded Shadow Maps. 2007.

37. Van J.M.P., Software W.-I., Hart -Nvidia E. Using Virtual Texturing to Handle Massive Using Virtual Texturing to Handle Massive Texture Data Texture Data. 2010.

38. Multi-adapter systems and Direct3D 12; Shared resources across adapters // Microsoft Learn. URL: https://learn.microsoft.com/en-us/windows/win32/direct3d12/multi-engine (дата обращения: 01.09.2025).