УДК 519.674
https://doi.org/10.47533/2020.1606-146X.05

***М. Б.*[*]*МУСТАФИН*[1], *О. Н. ТҰРАР*[1], *Д. Ж. АХМЕД-ЗАКИ*[2]**

[1]Казахский национальный университет им. аль-Фараби;
[2]Университет Международного Бизнеса

ТЕСТИРОВАНИЕ ВИЗУАЛИЗАЦИИ VULKAN ДЛЯ ГЕОМОДЕЛЕЙ НА СИСТЕМАХ С ГРАФИЧЕСКИМИ ПРОЦЕССОРАМИ ДЛЯ ТРАССИРОВКИ ЛУЧЕЙ

В данной работе описывается разработка высокопроизводительного приложения для представления двумерных и трехмерных результатов численных вычислений в графическом виде. Сравниваются два модуля визуализации, разработанные на основе разных методов перевода трехмерной модели в двумерное изображение. Для визуализации моделей используются технологии Vulkan и алгоритмы растеризации и Ray Tracing, и проводятся сравнительные анализы работы двух приложений в разных условиях. Разработанные модули визуализации можно использовать для отображения результатов численного математического моделирования на трехмерных сетках.

***Ключевые слова*:** *Vulkan, 2D, 3D, компьютерная графика, визуализация, сеточная модель, трассировка лучей.*

ВВЕДЕНИЕ. Разработка систем визуализации двумерных и трехмерных данных для анализа результатов численного моделирования является одним из развивающихся и важных направлений. Представление результатов в графическом виде улучшает восприятие и использование специалистами. В связи с использованием новейших технологий разработано приложение для визуализации результатов вычислений в реальном времени.

В этой области существует высокая потребность в удобных методах представления данных, что привело к развитию индустрии специализированного программного обеспечения для визуализации результатов численных расчетов различными способами. В отличие от большинства других областей научной визуализации, визуализация на нефтяных и газовых месторождениях реализует классическую полигональную визуализацию для представления геометрической модели месторождения. Геологические модели нефтяных месторождений обычно слишком велики с точки зрения количества элементов, чтобы их можно было легко визуализировать со стабильной частотой кадров. Проблемы были связаны с тем, что вычисления ведутся на суперкомпьютерных системах, а визуализация работает на клиентском устройстве. В последнее время были представлены определенные способы визуализации очень большого количества полигонов [1].

Для визуализации моделей на компьютерах была использована технология Vulkan. На сегодняшний день практически все устройства (ПК, мобильные телефоны, планшеты) поддерживают технологию Vulkan. Главным достоинством низкоуровнего Vulkan API является прямой доступ к аппаратным ресурсам графического процессора и распределение нагрузки между центральным процессором и графическим процессором. Таким образом, можно добиться высокой производительности и долгой автономности для мобильных устройств.

Используя технологию Vulkan, были разработаны модули 3D визуализации для настольных компьютеров, оснащенных дискретной графической картой. Также раз-

*Адрес для переписки. E-mail:mustafin.mb@gmail.com

работан модуль 3D визуализации с использованием новейшей технологии Vulkan Ray Tracing в реальном времени. Модуль визуализации, разработанный с использованием технологии трассировки лучей, визуализирует различные модели в реальном времени, что позволяет визуально наблюдать за изменениями какого - либо процесса.

Также в статье представлены примеры визуализаций разных моделей компьютерах. Отметим, что разработанное приложение может визуализировать любые результаты численного математического моделирования на структурированных и неструктурированных 3D сетках.

ОБЗОР ЛИТЕРАТУРЫ. На сегодняшний день существуют несколько программных интерфейсов, использующие двумерную и трехмерную компьютерную графику, такие как OpenGL, DirectX и Vulkan. OpenGL является высокоуровневым программным интерфейсом для графических устройств [2-4]. API Vulkan является приемником OpenGL, но они очень сильно отличаются. Со спецификациями Vulkan API можно ознакомится в литературах [5-8]. Vulkan поддерживает единственный шейдерный язык SPIR-V [9].

В исследовательской работе [10] изучаются высокопроизводительные API, и проводятся подробные сравнения между стандартным OpenGL и новейшими API таких как Vulkan и DirectX 12. Основное отличие технологии Vulkan от DirectX 12 в кроссплатформенности. DirectX 12 доступен только для ПК с ОС Windows 10, тогда как Vulkan доступен практически для всех устройств (ПК, мобильные телефоны, планшеты).

В статье описывается разработка приложения для визуализации нефтяных месторождений на настольных платформах с использованием технологии Vulkan и проводится тщательный анализ производительности при визуализации больших моделей.

В работе [11,12] разработаны оптимизированные алгоритмы визуализации результатов численного математического моделирования на структурированных сетках с использованием языка шейдеров библиотеки OpenGL, тогда как в описываемой работе используется технология Vulkan и готовый модуль визуализирует любые результаты численного математического моделирования на структурированных и неструктурированных сетках, а для повышения производительности используется технология Vulkan Real Time Ray Tracing [13,14]. Как и в большинстве подобных публикаций, эффективность метода визуализации измеряется путем сравнения количества визуализируемых кадров в секунду (FPS) с использованием различных технологий.

Кроме того, необходимость высокопроизводительной визуализации результатов расчетов нефтегазовых месторождений связана с наличием высокопроизводительных вычислений таких задач. Подобные исследования представлены в предыдущих работах авторов [15-17].

МЕТОДЫ.

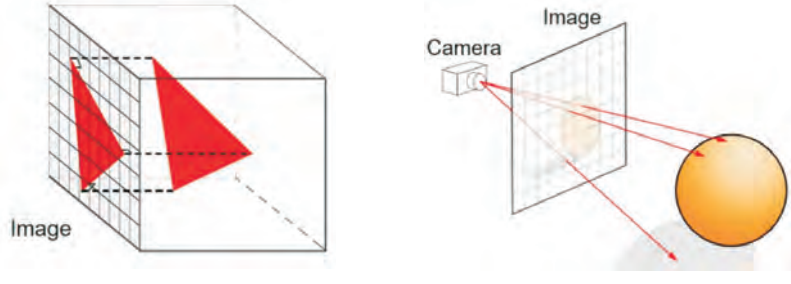
Постановка задачи. Чтобы протестировать и сравнить несколько модулей визуализации, нам необходимо определить метод измерения эффективности визуализации. Мы попытались добиться плавных трансформационных манипуляций геометрической модели нефтяного месторождения. Плавность анимации определяется частотой кадров, представленных на экране в секунду (FPS). Комфортное и удобное количество кадров в секунду может варьироваться для разных людей, но в большинстве

случаев оно должно быть ниже 40 в научных приложениях. Некоторые пользователи предпочитают значения 60 или выше. В данной работе мы рассмотрим, сравним и проанализируем на основе гораздо более высоких значений FPS. Это связано с тем, что дальнейшее увеличение геометрической модели приведет к пропорциональному изменению количества FPS.

Для достижения максимально возможных результатов визуализации с помощью разработанных модулей мы использовали сгенерированные модели на основе простой геометрии куба.

Технологии визулизации. Существует два метода перевода 3D модели в двумерное изображение для отображения на экране: растеризация 3D модели и трассировка лучей. Обычно используется метод растеризации. Метод растеризации заключается в следующем. На плоскость экрана проецируются примитивы 3D модели и с помощью специальных алгоритмов получается цвет каждого пикселя, лежащий внутри спроецированных примитивов.

Трассировка лучей - это метод создания трехмерных моделей с применением принципа, подобного реальным физическим процессам. От камеры для каждого пикселя экрана генерируется луч до объекта, затем пиксель окрашивается цветом в месте пересечения с объектом луча. Пример отрисовки методом растеризации и трассировки лучей можно увидеть на рисунке 1.



А) Б)

Рисунок 1 – А) отрисовка методом растеризации,
Б) отрисовка методом трассировки лучей

Для работы с трассировкой лучей существуют следующие элементы и понятия:

- Структуры ускорения (Acceleration structures) - специальный объект, инкапсулирующий внутреннюю картину геометрии. Его можно рассматривать как один из видов дерева (BVH), ускоряющий поиск пересечений луча и геометрии.
- Таблица привязки шейдеров (Shader Binding Table, SBT) - структура данных, позволяющая API отправить несколько шейдеров (и / или его отдельных стадий) для трассировки лучей, а затем в динамическом виде вызывают шейдер из этой таблицы в шейдерах.
- Новая команда, которая запускает трассировку (vkCmdTraceRaysNV).
- Новый конвейер (pipeline), который может работать с таблицей шейдеров для трассировки лучей.

Компьютерное приложение было реализовано с использованием новейших расширений для трассировки лучей через API Vulkan

Vulkan – это API для графических вычислительных устройств. Vulkan является кроссплатформенным API для высокопроизводительного доступа к графике и вычислений на современных видеокартах. Он может использоваться для разнородных устройств, таких как графические процессоры, мобильные устройства, планшеты. Vulkan предоставляет приложениям непосредственное управление графическим ускорением для улучшения эффективности и производительности. Vulkan единственный высокопроизводительный графический API работающий с несколькими операционными системами включая Windows, Linux и Android.

Входными данными визуализации являются результаты расчетов различных задач. Они содержат геометрию и разные характеристики пласта и имеют формат GRDECL. На рисунке 2 демонстрируется приложения визуализации для мобильных и настольных устройств.

Модуль визуализаций, разработанные с использованием Vulkan API для настольных систем, визуализирует различные модели, что позволяет визуально наблюдать за процессами.

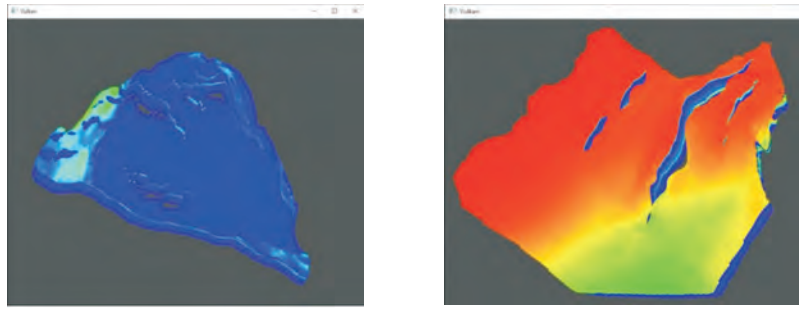


Рисунок 2 – приложение для настольных устройств.

РЕЗУЛЬТАТЫ. Для проведения тестов с визуализатором на компьютерах был использован персональный компьютер (Core i7 3770 3.40 GHz, 16Gb DDR3), оснащенный дискретной графической картой (nVidia GeForce RTX2080 ti, 11Gb GDDR6). На таблице 1 описаны характеристики использованных устройств для тестирования.

Таблица 1 – Характеристики графического устройства Geforce 2080 ti.

	Nvidia Geforce 2080 ti
Количество вычислительных блоков	4352
Теоретическая производительность	28,5/14,2 TFlops
Интерфейс памяти	352 бит
Скорость передачи данных памяти	14000 МГц
Пропускная способность памяти	448 ГБ/с
Выделенная видеопамять	8 ГБ GDDR6

Сравнение количества кадров в секунду при визуализации очень больших двумерных сеток с использованием Vulkan методом растеризации и трассировки лучей показано в таблице 2.

Таблица 2 – Количество кадров в секунду при визуализации различных сеток большого размера.

Размер	Количество ячеек	Vulkan растеррлеу (FPS)	Vulkan RT (FPS)
100x100	20,000	3500	2600
500x500	500,000	3600	1285
1000x1000	2,000,000	1650	1095
2000x2000	8,000,000	650	1005
3000x3000	18,000,000	265	585
4000x4000	32,000,000	155	625
5000x5000	50,000,000	100	600
5700x5700	64,980,000	78	610

Каждая ячейка сетки состоит из двух треугольников, поэтому количество полигонов в два раза больше количества ячеек.

Как видно в таблице 2, при визуализации с использованием метода растеризации в небольших размерах, т. е. примерно до 2000x2000 или до 8 000 000 узлов, количество кадров в секунду больше, чем количество кадров в секунду при визуализации с использованием метода трассировки лучей. Однако, по мере увеличения размера модели результаты при визуализации с использованием метода трассировки лучей превосходят результаты метода растеризации. Метод растеризации, показанный на графике на рисунке 3, показывает, что результат уменьшается с увеличением размера, в то время как метод трассировки лучей показывает стабильный результат при размере более чем 18 миллионов узлов.

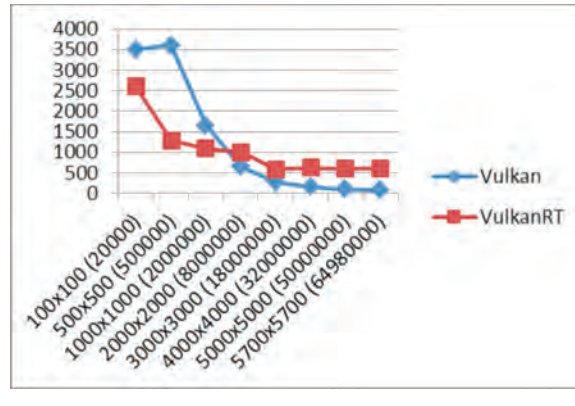


Рисунок 3 – Относительный график количества кадров в секунду при использовании методов растеризации и трассировки лучей

Кроме того, чем больше сеточная модель занимает область окна программы, т. е. чем ближе модель к экрану, тем больше она влияет на производительность, и разница между двумя методами меняется. Результаты визуализации приближенной модели с использованием размеров в таблице 2 показаны в таблице 3.

Таблица 3 – Количество кадров в секунду при визуализации различных больших сеток приближенной модели

Размер	Количество ячеек	Vulkan растеризация (FPS)	Vulkan RT (FPS)
100x100	20000	3500	2340
500x500	500000	3600	1000
1000x1000	2000000	1660	400
2000x2000	8000000	660	207
3000x3000	18000000	265	166
4000x4000	32000000	157	166
5000x5000	50000000	102	149
5700x5700	64980000	77	200

Как видно из таблицы 3, метод растеризации работает более эффективно, чем метод трассировки лучей, до 4000x4000, но при больших размерах метод трассировки лучей показывает стабильный результат, и количество кадров в секунду метода растеризации снижается. Причина уменьшения результата метода растеризации заключается в том, что во время растеризации проекция каждого узла проецируется на плоскость экрана и закрашиваются пиксели, поэтому с увеличением количества узлов увеличивается число процессов растеризации. Метод трассировки лучей генерирует лучи через плоскость экрана, поэтому число кадров в секунду уменьшается по мере приближения модели к окну программы, но график на рисунке 4 показывает, что с большими сетками трассировка лучей работает лучше, чем растеризация.

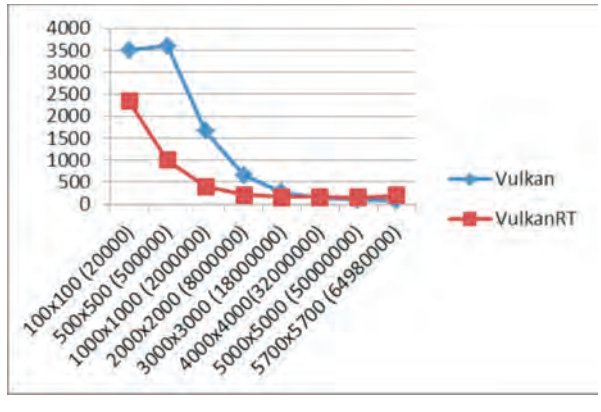


Рисунок 4 – График количества кадров в секунду при визуализации различных больших сеток приближенной модели.

Кроме того, размерность модели сетки влияет на производительность, и разница между этими двумя методами также изменяется. Таблица 4 показывает сравнение количества кадров в секунду двух методов при визуализации трехмерных сеток. Для сравнения была использована модель куба с различными размерами и заполненные случайными цветами изображенные на рисунке 5.

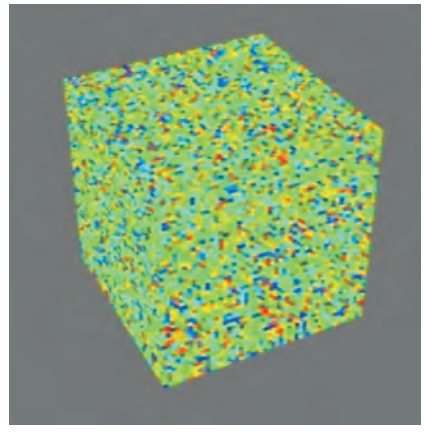


Рисунок 5 – Трехмерная модель куба.

Таблица 4 – Количество кадров в секунду при визуализации различных трехмерных сеточных моделей

Размер	Количество ячеек	Vulkan растеризация (FPS)	VulkanRT
10x10x10	2000	3400	2450
20x20x20	16000	3600	2350
50x50x50	250000	3300	2240
100x100x100	2000000	1460	1700
150x150x150	6750000	515	1550
200x200x200	16000000	255	1325
250x250x250	31250000	146	1120
280x280x280	43904000	109	1000
300x300x300	54000000	88	900

Как видно из таблицы 4, метод трассировки лучей показывает в несколько раз более высокую производительность, чем метод растеризации при визуализации моделей размером более 100x100x100. Поскольку в случае растеризации учитываются все узлы модели, а в случае с трассировки лучей луч возвращает только цвет на пересечении, а невидимые слои позади не учитываются, поэтому для трехмерных моделей метод трассировки лучей дает хорошую производительность. Результаты таблицы 4 показаны на графике на рисунке 6.

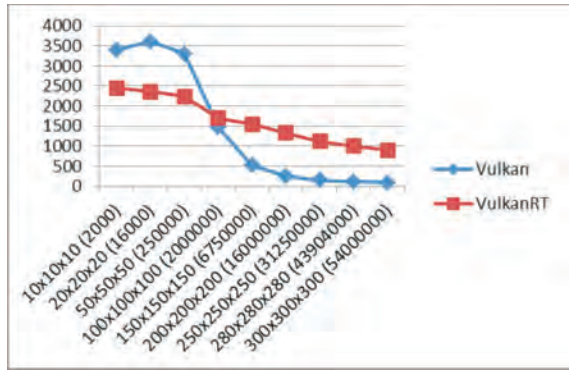


Рисунок 6 – График количества кадров в секунду при визуализации трехмерной сеточной модели.

Результаты визуализации приближенной трехмерной модели показаны в таблице 5 и на графике на рисунке 7.

Таблица 5 – Количество кадров в секунду при визуализации различных больших сеток приближенной трехмерной модели

Размер	Количество ячеек	Vulkan растеризация (FPS)	VulkanRT
10x10x10	2000	3350	2160
20x20x20	16000	3500	2200
50x50x50	250000	1840	1980
100x100x100	2000000	770	1690
150x150x150	6750000	358	1410
200x200x200	16000000	200	1270
250x250x250	31250000	120	1230
280x280x280	43904000	88	1130
300x300x300	54000000	75	1060

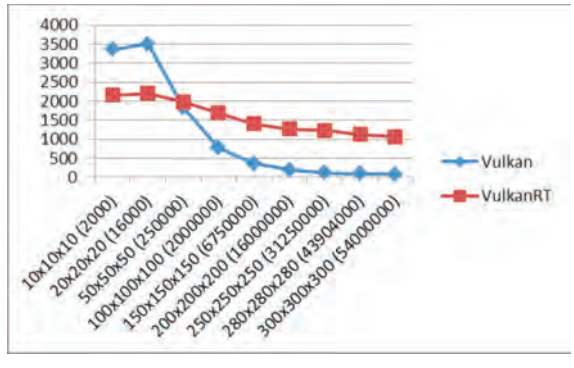


Рисунок 7 – График количества кадров в секунду при визуализации различных больших сеток приближенной трехмерной модели

АНАЛИЗ. Анализируя результаты, полученные при сравнении модулей визуализации, результаты, полученные на настольном GPU, очень высокие и начинается от 4300 FPS при работе с моделями малого размера и с увеличением размера снижается до 615 FPS.

На таблице 2 показаны результаты FPS при визуализации двумерной модели на настольном GPU с использованием метода растеризации и трассировки лучей. Отрисовка методом растеризации до размера 2000x2000 показывает 1650 кадров в секунду, а в методе трассировки лучей FPS равен 1095. По мере увеличения размера модели FPS отрисовки методом трассировки лучей превосходят результаты метода растеризации. На таблице 3 можно увидеть результаты FPS приближенной двумерной модели. При отрисовке приближенной модели до размера 3000x3000 можно заметить, что метод растеризации показывает больше FPS, чем метод трассировки лучей, но при больших размерах ячеек количество кадров в секунду при отрисовке методом растеризации ниже, чем количество кадров в секунду при отрисовке методом трассировки лучей.

На таблице 4 показаны результаты FPS при визуализации трехмерной модели на настольном GPU с использованием метода растеризации и трассировки лучей. Из таблицы можно заметить, что отрисовка методом трассировки лучей показывает более высокий FPS, чем метод растеризации при отрисовке модели размером более 100x100x100. На таблице 5 можно увидеть результаты FPS приближенной трехмерной модели. Тут можно заметить, что, начиная с размера 50x50x50 трехмерной модели, количество кадров в секунду при отрисовке методом трассировки лучей в несколько раз превосходит количества кадров в секунду при отрисовке методом растеризации.

ОБСУЖДЕНИЕ. В результате сравнения метода растеризации показал хорошую производительность при небольших размерах, но по мере увеличения размера модели сетки метод трассировки лучей показал стабильную и высокую производительность. Однако по мере приближения к модели вы можете видеть, что количество кадров в секунду значительно изменяется в методе трассировки лучей. Это связано с тем, что площадь искомого луча увеличивается, а метод растеризации не меняет количество операций, поэтому он существенно не меняется, поскольку проецируется каждая ячейка на экран. Кроме того, при визуализации трехмерных сеточных моделей метод трассировки лучей показал более высокую производительность, чем метод растеризации. Это объясняется тем, что метод растеризации вычисляет проекции каждой ячейки на плоскость экрана, а метод трассировки лучей не учитывает невидимые участки объекта. Следовательно, для визуализации трехмерных сеточных моделей, выгодно использовать метод трассировки лучей.

ЗАКЛЮЧЕНИЕ. В данной статье была поставлена задача разработки высокопроизводительного компьютерного приложения. Получены результаты работы комьютерного приложения с использованием метода расеризации и метода трассировки лучей. В качестве входных данных для тестирования производительности была использована модель куба разного размера. Были проведены сравнительные анализы результатов комьютерного приложения с использованием двух методов отображения.

ЛИТЕРАТУРА

1 A first look at Unreal Engine 5, 2020 - https://www.unrealengine.com/en-US/blog/a-first-lookat-unreal-engine-5

2 Mark S., Kurt A. “The OpenGL Graphics System: A Specification.” (2008).

3 Sellers G., Wright R.S. Jr., Haemel N. OpenGL SuperBible: Comprehensive Tutorial and Reference (6th Edition). Addison Wesley Professional; 6 edition (July 31, 2013). 2013. – P. – 848.

4 OpenGL programming guide: the official guide to learning OpenGL, version 4.3 / Dave Shreiner, Graham Sellers, John Kessenich, Bill Licea-Kane ; the Khronos OpenGL ARB Working Group.---Eighth edition.

5 Селлерс Г. Vulkan. Руководство разработчика. Официальное руководство / пер. С англ. А. Б. Борескова. – М.:ДМК Пресс, 2017. – 394с.

6 Khronos Vulkan Working Group. Vulkan 1.0.98 - A Specification (with KHR extensions). 1.0.98. Jan. 2019.

7 Pawel Lapinski, Vulkan Coockbook. Work through recipes to unlock the full potential of the next generation graphics API—Vulkan. Packt Publishing Ltd. Birmingham (2017).

8 Khronos Group. Vulkan. https://www.khronos.org/vulkan/. Accessed: 2017-11-13.

9 Khronos Group. SPIR Overview. https : / / www . khronos . org / spir/. Accessed: 2018-03-03.

10 Shiraef, Joseph A.. “An exploratory study of high performance graphics application programming interfaces.” (2016).

11 Бадретдинов М. Р., Бадретдинов Т. Р., Борщук М. С., Применение библиотеки opengl для визуализации результатов численного математического моделирования на сетках большой размерности. Вестник УГАТУ, 2015. – № 4. С. 84-94 .[Badretdinov M. R., Badretdinov T. R., Borshchuk M. S., Primenenie biblioteki opengl dlya vizualizacii rezul'tatov chislennogo matematicheskogo modelirovaniya na setkah bol'shoj razmernosti. Vestnik UGATU, 2015. – № 4. S. 84-94]

12 Abraham, Frederico & Celes, Waldemar. (2009). Distributed Visualization of Complex Black Oil Reservoir Models.. 87-94. 10.2312/EGPGV/EGPGV09/087-094.

13 Khronos Group. Ray Tracing In Vulkan. – https://www.khronos.org/blog/ray-tracing-invulkan#raytracing1a Accessed: 2020-03-25.

14 NVIDIA Vulkan Ray Tracing Tutorial – https://developer.nvidia.com/rtx/raytracing/vkray Accessed: 15.02.2019.

15 D.Zh. Akhmed-Zaki, T.S. Imankulov, B. Matkerim, B.S. Daribayev, K.A. Aidarov, O.N. Turar. Large-scale simulation of oil recovery by surfactant-polymer flooding. Eurasian Journal of mathematical and computer applications. – 2016. V. 4, – P. 12-31.

16 Akhmed-Zaki D.Zh., Daribayev B.S., Imankulov T.S., Turar O.N. High-performance computing of oil recovery problem on a mobile platform using CUDA technology. Eurasian Journal of mathematical and computer applications. – 2017. V. 5, – P. 4-13.

17 T.S. Imankulov, D.Zh. Akhmed-Zaki, B.S. Daribayev and O.N. Turar. HPC Mobile Platform for Solving Oil Recovery Problem. Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016), V. 2 Lisbon, Portugal. 29 - 31 July 2016. – P. 595-598.

18 Клочков Михаил Аркадьевич. “К решению задачи визуализации результатов моделирования процессов разработки нефтегазовых месторождений” Известия Института математики и информатики Удмуртского государственного университета, 2017. – V. 49. – P. 3-16. [Klochkov Mihail Arkad'evich. “K resheniyu zadachi vizualizacii rezul'tatov modelirovaniya processov razrabotki neftegazovyh mestorozhdenij” Izvestiya Instituta matematiki i informatiki Udmurtskogo gosudarstvennogo universiteta, 2017. – V. 49. – R. 3-16.]

19 Геологическое и гидродинамическое моделирование месторождений нефти и газа: учебное пособие / Е.А. Гладков; Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2012. – 99 с. [Geologicheskoe i gidrodinamicheskoe modelirovanie

mestorozhdenij nefti i gaza: uchebnoe posobie / E.A. Gladkov; Tomskij politekhnicheskij universitet. – Tomsk: Izd-vo Tomskogo politekhnicheskogo universiteta, 2012. – 99 s.]

20 Mullen, Tim R., Christian Kothe, Yu M. Chi, Alejandro Ojeda, Trevor Kerth, Scott Makeig, Gert Cauwenberghs and Tzyy-Ping Jung. “Real-time modeling and 3D visualization of source dynamics and connectivity using wearable EEG.” 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) (2013): 2184-2187.

М. Б.МУСТАФИН[1]; О. Н. ТҰРАР[2], Д. Ж. АХМЕД-ЗАКИ[2]

[1]әл-Фараби атындағы Қазақ ұлттық университеті, Алматы, Қазақстан
[2]Халықаралық бизнес университеті, Алматы, Қазақстан

ГРАФИКАЛЫҚ ПРОЦЕССОРЛАРЫ БАР ЖҮЙЕЛЕРДЕ СӘУЛЕЛЕРДІҢ ЖОЛ ТАРТУЫН БАҚЫЛАУ ҮШІН VULKAN ВИЗУАЛИЗАЦИЯСЫН ТЕСТІЛЕУ

Бұл жұмыста графикалық түрде сандық есептеулердің екі өлшемді және үш өлшемді нәтижелерін ұсынуға арналған жоғары өнімді қосымшаның әзірленуі сипатталған. Үш өлшемді модельді екі өлшемді бейнеге аударудың әртүрлі әдістеріне негізделген екі визуализациялау модулі салыстырылады. Модельдерді визуализациялау үшін Vulkan технологиялары және растеризация және Ray Tracing алгоритмдері қолданылады, және әртүрлі жағдайларда екі қосымшаның жұмысына салыстырмалы талдау жасалады. Әзірленген визуализациялау модульдерін сандық математикалық модельдеу нәтижелерін үш өлшемді торларда көрсету үшін қолдануға болады.

***Түйін сөздер**: Vulkan, 2D, 3D, компьютерлік графика, визуализация, торлы модель, сәулелік іздестіру.*

M. B. MUSTAFIN[1], O. N. TURAR[1], D. ZH. AKHMED-ZAKI[2]

[1]Al-Farabi Kazakh national University, Almaty, Kazakhstan
[2]University of International Business, Almaty, Kazakhstan

TESTING VULKAN VISUALIZATION FOR GEOMODELS ON SYSTEMS WITH GRAPHIC PROCESSORS FOR RAY TRACING

This paper describes the development of a high-performance application for representing two-dimensional and three-dimensional results of numerical calculations in graphical form. Two visualization modules developed on the basis of different methods for translating a three-dimensional model into a two-dimensional image are compared. Vulcan technologies and rasterization and Ray Tracing algorithms are used for model visualization, and comparative analyses of the operation of two applications in different conditions are performed. The developed visualization modules can be used to display the results of numerical mathematical modeling on three-dimensional grids.

***Key words**: Vulkan, 2D, 3D, computer graphics, visualization, grid model, ray tracing.*