# Реализация системы частиц в реальном времени на GPU в Unity3D

## Аннотация

Все стандартные системы частиц в Unity используют CPU, что сильно загружает центральный процессор при ресурсоёмких проектах. Из-за этого большинство приложений невозможно использовать на слабых компьютерах. Создание системы частиц на GPU позволит снизить нагрузку с работы CPU, что позволит создавать более качественную картинку, распределяя нагрузку между процессорами. Это позволит увеличить быстродействие и оптимизировать систему отображения.

Реализация данного метода позволяет сравнить возможности ранее реализованных систем частиц на CPU с возможностями GPU, оценив достоинства и недостатки существующих решений. На данный момент реализация находится на ранних этапах разработки из-за отсутствия существующих решений, использующих непосредственно GPU в Unity. В рамках работы будут сравнены минусы и плюсы различных существующих решений на GPU при реализации системы частиц, а так же принятые по ходу анализа решения для последующей реализации.

## Введение

С развитием технологий, использующих визуальную составляющую, где упор идет на качество отображения и быстродействие системы, встает вопрос об оптимизации имеющихся технологий, усложнения архитектуры центральных и графических процессоров. В то время, как оптимизировать центральный процессор становится всё сложнее, графический процессор не имеет подобных проблем. Одна из ведущих компаний отмечает стремительный рост производительности GPU, позволяющий создавать более сложные приложения на базе ИИ. Основой в компьютерной графике также служит визуальное отображение качественной картинки без потери быстродействия, а создание системы частиц в реальном времени на GPU позволяет снизить нагрузку с работы CPU при ресурсоёмком проекте. Для последующей работы необходимо провести:

* Анализ существующих решений на GPU и CPU;
* Сбор и анализ ключевых требований;
* Проектирование системы частиц в реальном времени на GPU;
* Анализ быстродействия и гибкости в сравнении с существующими GPU и CPU аналогами;
* Выбор методов и решений для улучшения работы системы частиц.

## Обзор предметной области

В настоящее время рост частот универсальных процессоров упёрся в физические ограничения и высокое энергопотребление, и увеличение их производительности всё чаще происходит за счёт размещения нескольких ядер в одном чипе. Продаваемые сейчас процессоры предназначены для обычных приложений, используют множественный поток команд и данных. Каждое ядро работает отдельно от остальных, исполняя разные инструкции для разных процессов. В это время можно сказать, что графический процессор данных сложностей особо не испытывает и может использовать несколько тысяч ядер для параллельной обработки изображения. Но графический процессор может выполнить лишь часть операций, которые может выполнить центральный процессор, хотя он делает это с невероятной скоростью. GPU будет использовать сотни ядер, чтобы выполнить срочные вычисления для тысяч пикселей и отобразить при этом сложную 3D графику. Но для достижения высоких скоростей GPU должен выполнять однообразные операции. Центральные же процессоры имеют больший набор инструкций, поэтому они могут выполнять более широкий диапазон функций. Также CPU работают на более высоких максимальных тактовых частотах.

На данный момент аналоги для Unity с использованием только GPU можно найти на аккаунтах github непосредственно у разработчиков Unity в качестве наработок на будущее. Другие же примеры для сравнения используют только алгоритмы для передачи работы от CPU к GPU при рендере системы частиц.

# [KvantSpray](https://github.com/keijiro/KvantSpray)

Эффекты KvantSPray требуют текстуры HDR с плавающей точкой для сохранения состояния анимации. Большинство мобильных устройств в настоящее время не соответствуют этому требованию.

# SprayMV

Исходная версия SprayMV не поддерживает векторы движения и, следовательно, не может быть хорошо использована для эффектов пост-обработки, требующих векторов движения (например, размытия движения). SprayMV генерирует точные пиксельные векторы движения от каждой отдельной частицы. Это позволяет изобразить движение сложной системы частиц очень естественным образом.

# PopcornFX

Программа реализации системы частиц, с возможностью гибкой настройки поведения частиц и возможностью последующего портирования в другие среды разработки.

### Критерии сравнения аналогов

#### Глубина интеграции в Unity

Использование внешней библиотеки для выполнения поставленной задачи. Данный критерий оценивается по тому, подключается ли внешняя библиотека в Unity или прописан весь алгоритм в самой среде.

#### Возможности настройки

Данный критерий описывает можно ли задавать количество частиц, их скорость, настройки отображения, а так же удобство их изменения.

#### Поддержка MotionVectors и VectorFields

Данный критерий описывает возможность использования в системе частиц MotionVectors и вектора полей.

### Таблица сравнения по критериям

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерий** | [**KvantSpray**](https://github.com/keijiro/KvantSpray) | **SprayMV** | **PopcornFX** |
| Глубина интеграции в Unity | - | - | + |
| Возможности настройки | + | + | + |
| Поддержка MotionVectors и VectorFields | + | - | + |

### Выводы по итогам сравнения

По результатам сравнения программа PopcornFX имеет очевидные преимущества перед другими аналогами. Она имеет возможность создавать, настраивать систему частиц для работы на GPU с последующим портированием готовой системы частиц на Unity.

## Выбор метода решения

Решение представляет собой реализацию системы частиц схожую с реализацией в PopcornFX для Unity, её полную настройку для работы с использованием GPU. Текущими требованиями служат:

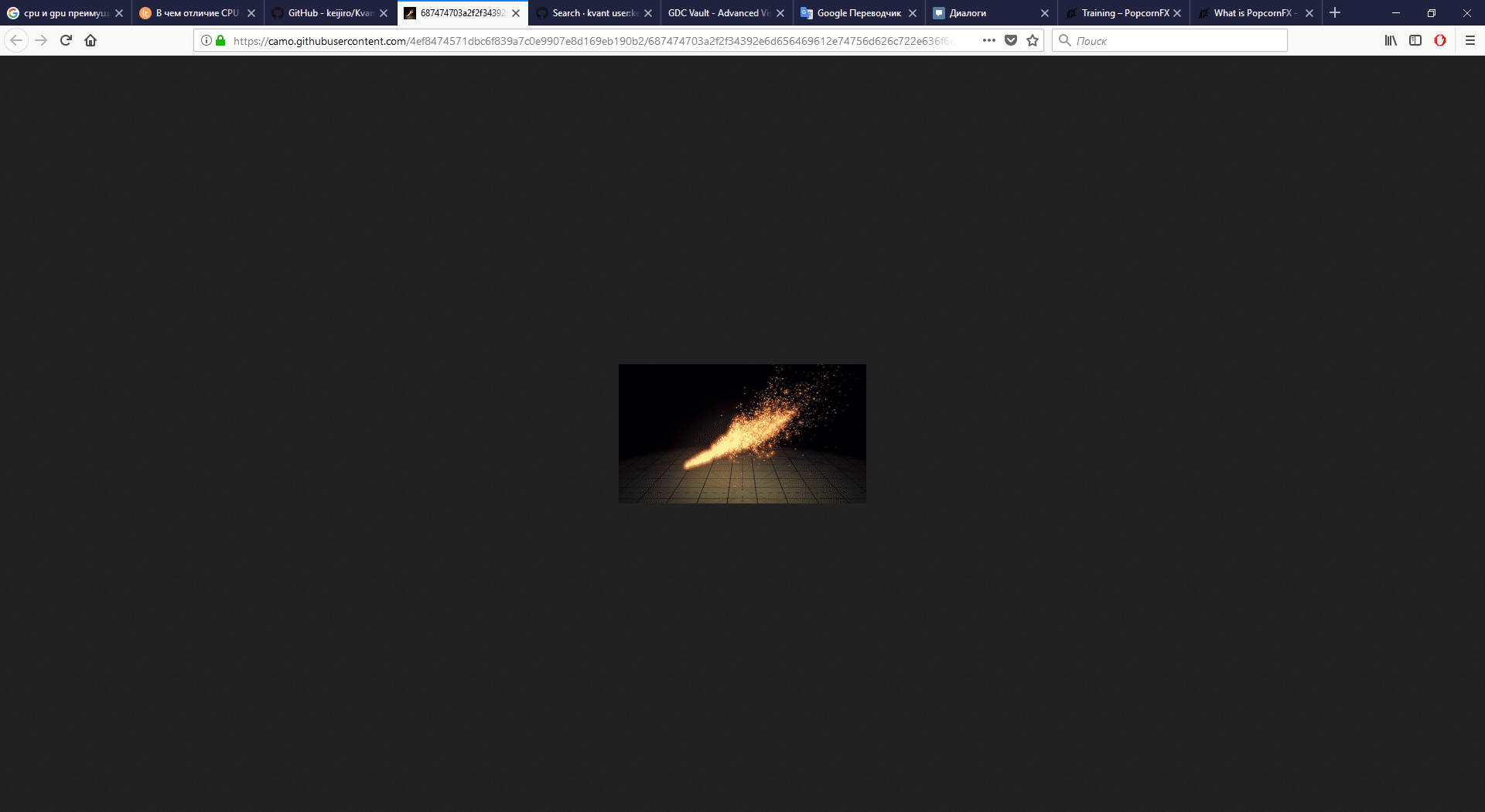
* Поддержка MotionVectors, чтобы движущиеся векторы отслеживали скорость каждого пикселя объекта от одного кадра к другому. Используя эту информацию, можно применять определенные эффекты изображения, такие как размытость изображения или временное сглаживание.
* Поддержка вектора полей, благодаря которой можно влиять на движение частиц.
* Возможности детальной настройки системы частиц непосредственно в самом Unity.

## Описание решения

Детально рассмотрим существующие решения для реализации требований.

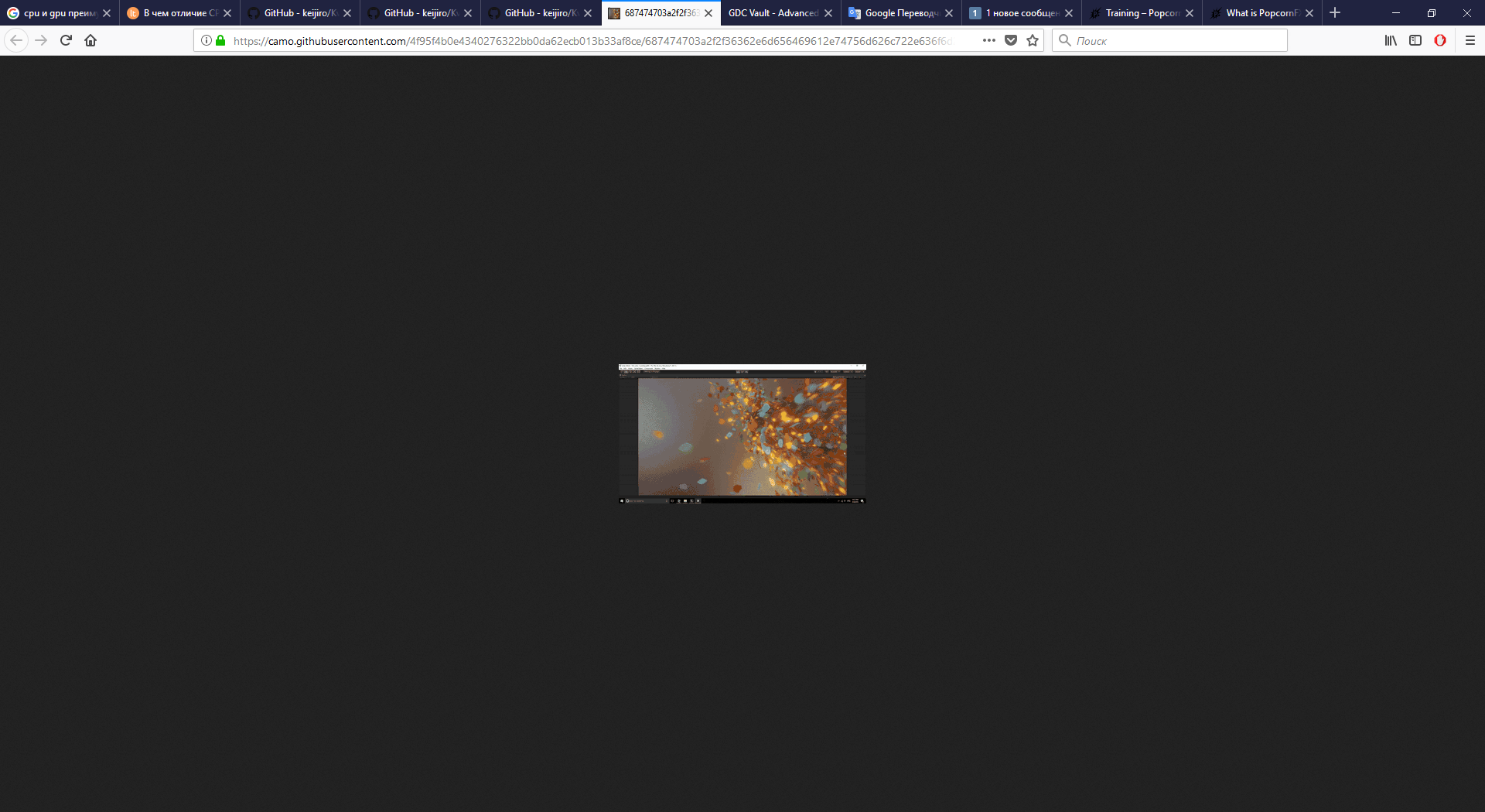
# [KvantSpray](https://github.com/keijiro/KvantSpray)

В данном решении можно изучить возможности работы с системой частиц непосредственно в рабочей среде Unity. Она не подключает внешние библиотеки для работы, а это означает что поведение и настройки системы частиц придётся выполнять вручную. Это задаёт гибкость при создании поведения системы частиц независимо от окружающих объектов. Но имеет и минус в работе с HDR изображениями.



# SprayMV

Так как SprayMV не поддерживает векторы движения, поведение системы частиц задаётся поведением окружающей среды, и тем самым позволяет изобразить движение сложной системы частиц очень естественным образом. С помощью этого спрея можно создавать сложные системы частиц, но по сути без возможности управления.



# PopcornFX

Эта программа позволяет как работать со сложными частицами, настраивая их поведение различными способами, так и имеет возможность дальнейшего портирование в различные среды разработки. Но недостатком является необходимость добавления сторонней библиотеки для работы созданной системы частиц.

## Заключение

При детальном анализе и изучении данных решений, можно сказать, что все они в какой-то мере имеют достоинства и недостатки для выполнения поставленной цели. Результатом работы можно считать интеграции идей SprayMV для возможности работы с системами частиц различной сложности и так же [KvantSpray](https://github.com/keijiro/KvantSpray) для возможности настройки системы частиц необходимым образом. Для анализа выполнения качества работы и последующих исправлениях будет создаваться аналогичная система в программе PopcornFX.

В дальнейших планах изучить большие варианты реализации, различные методы и алгоритмы, использующиеся в гибридных решениях CPU и GPU, а так же закончить систему частиц, отвечающую всем поставленным требованиям с анализом быстродействия в сравнении со схожими реализациями.

## Список литературы

1. Advanced Visual Effects with DirectX 11: Compute-Based GPU Particle Systems
2. KvantSpray [Электронный ресурс] URL: <https://github.com/keijiro/KvantSpray>
3. SprayMV [Электронный ресурс] URL: <https://github.com/keijiro/KvantSprayMV>
4. PopcornFX [Электронный ресурс] URL: <https://www.popcornfx.com/>
5. GPU Compute and Instanced Particles [Электронный ресурс] URL: http://zhengeng.net/gpuparticle