Разработка библиотеки для визуализации 3D графики

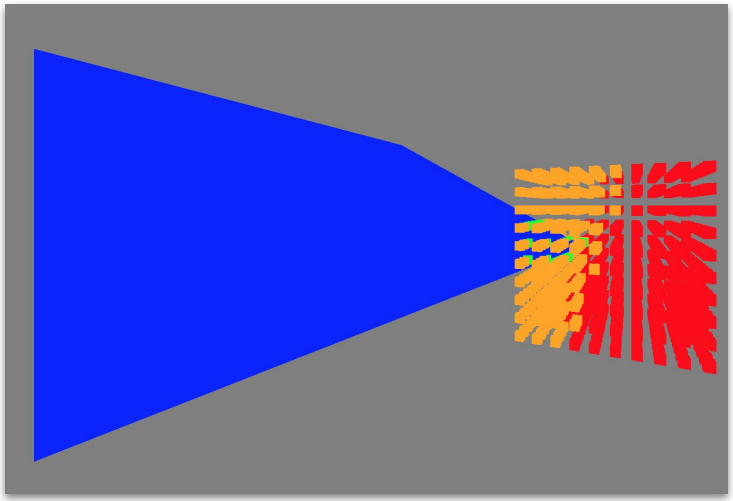
Курсовая работа

Орачев Егор Станиславович
группа 17.Б11-мм
СПбГУ

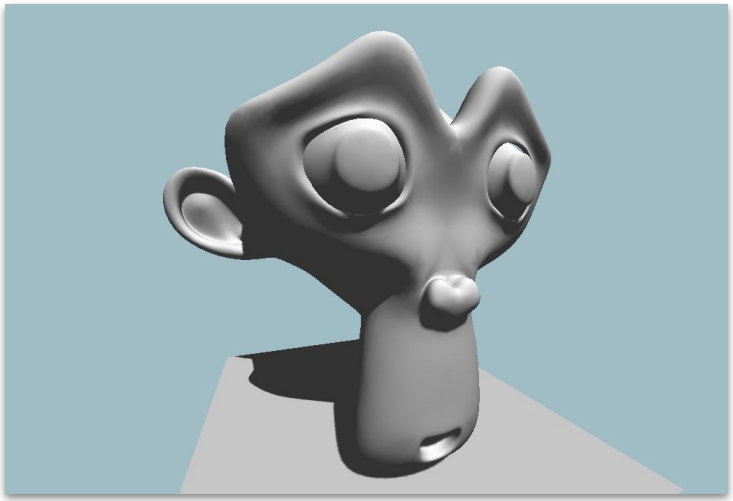
Научный руководитель:
ст.преп. А.А. Пименов

23 мая 2020 года

Применение



Отображение отладочной информации



Отображение 3D геометрии и моделей

Мотивация

- Развитие графического API Vulkan
- Расширение набора вычислительных платформ
- **Необходимость рефакторинга графической функциональности в библиотеке CoreCVS**

Постановка задачи

Цель работы — разработка библиотеки для визуализации 3D графики в реальном масштабе времени с использованием графического API Vulkan

Задачи:

- Исследование предметной области
- Разработка и формализация требований к проекту
- Разработка компонентов библиотеки
- Поддержка Vulkan API

Требования к проекту

- Язык разработки: C++11
- Система сборки: CMake
- Поддержка Windows, macOS, Linux
- Поддержка Vulkan API
- Поддержка экранных эффектов
- Независимость от оконной системы
- Независимость от графического API
- Независимость от структуры сцены

Существующие решения

На основе требований к проекту были выбраны и проанализированы следующие решения:

- Google ***Filament***
- Game Foundry ***BS Framework***
- Ogre Cave ***Ogre3D***

Существенный недостаток: сильная взаимосвязь между логикой отрисовки и системой управления специфическими форматами ресурсов

Структура библиотеки

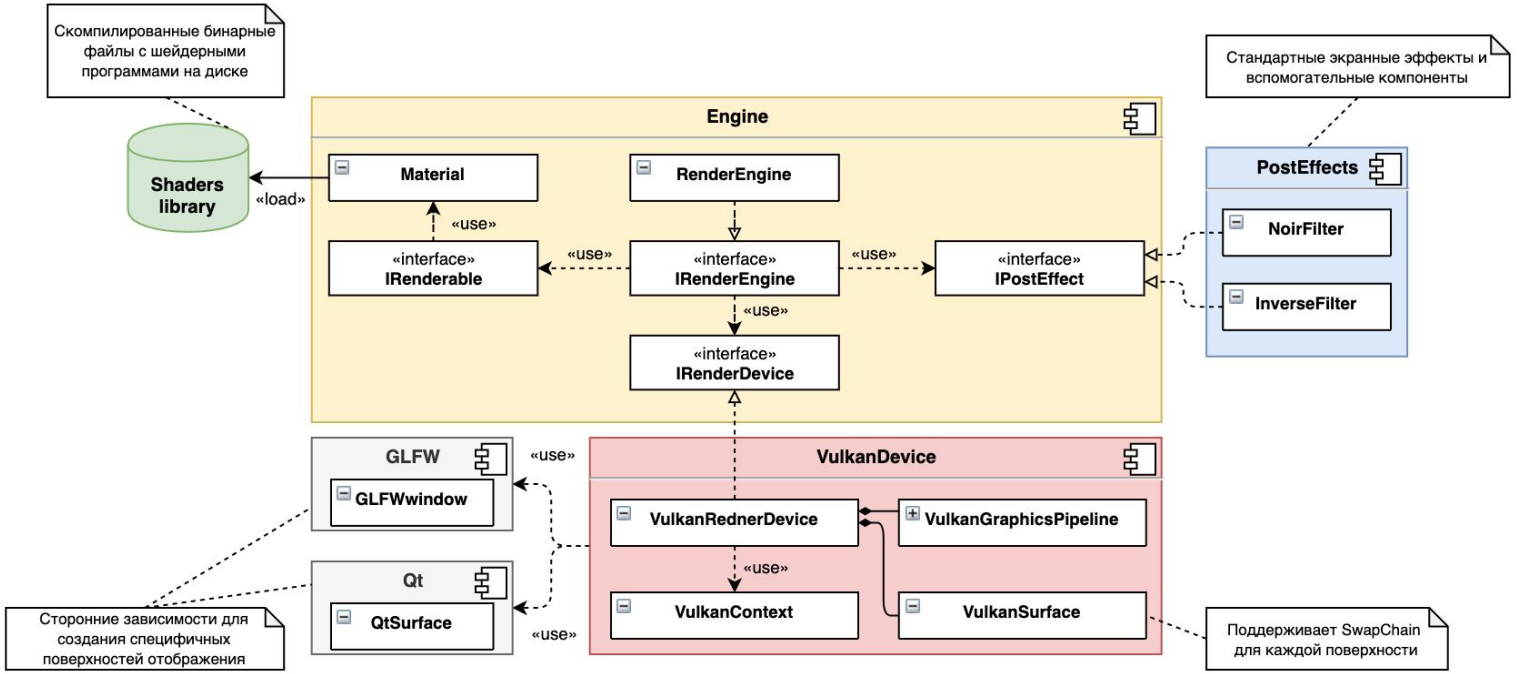


Диаграмма основных компонентов и классов разработанной графической библиотеки

Реализация

В рамках этой работы были выполнены следующие подзадачи:

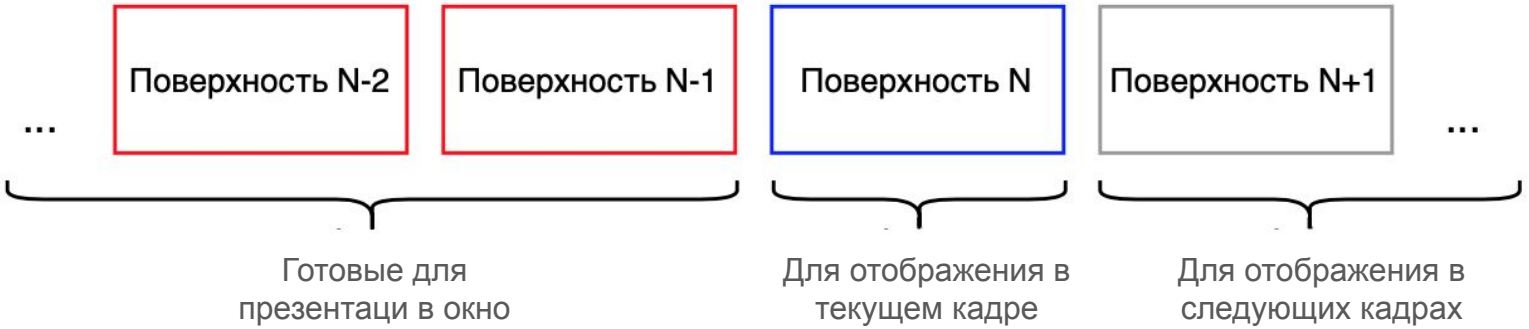
- Разработка компонентов *VulkanDevice*
- Разработка класса *Material*
- Разработка *RenderEngine*
- Поддержка экранных эффектов в *RenderEngine*
- Настройка непрерывной интеграции

Инициализация Vulkan девайса

- *VulkanContext* инкапсулирует состояние библиотеки
- Предоставляет доступ к абстрактному девайсу рендеринга:
  - Интегрированный графический процессор
  - Дискретный видеоадаптер
- Позволяет конфигурировать:
  - Расширения
  - Специфичные возможности девайса
  - Очереди обработки запросов

Отображение в нативное окно ОС

- *VulkanSwapchain* инкапсулирует логику презентации изображения, сгенерированного на видеокарте, в нативное окно ОС
- Создание поверхностей и обработка изменений размера окна
- Синхронизация чтения и отображения



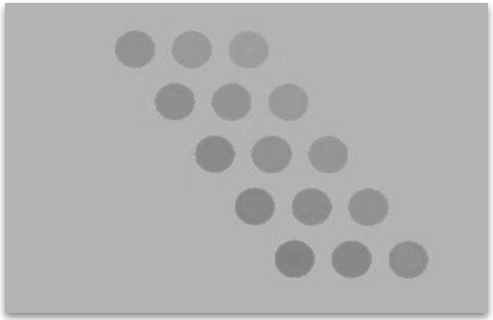
Конфигурация отображения

- *Material* включает в себя*:*
  - GPU программу
  - настройки конвейера
  - входные параметры программы
- Типобезопасный доступ к конфигурации
- Автоматическая упаковка данных
- *Минимизация* изменений состояния GPU

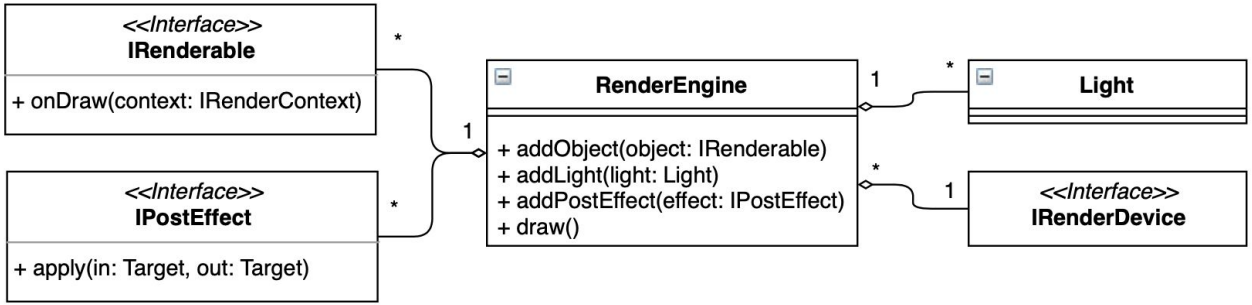
Физически корректное затенение на основе вариации модели Кука-Торренса

Система отображения

- *RenderEngine* поддерживает список доступных объектов, источников света и эффектов
- Генерацию карт теней
- Применение экранных эффекты к созданному изображению



Карта теней направленного источника света для сцены из 14 объектов



Экранные эффекты

- Применяются для обработки готового изображения
- Доступна информация о цвета и глубине для каждой точки

Черно-белый фильтр

Инвертированный цвет

Итоги

В рамках данной работы получены следующие результаты:

- Проведено исследование предметной области
- Разработаны и формализованы требования к проекту
- Разработаны соответствующие компоненты библиотеки
- Реализована поддержка Vulkan API

Репозиторий проекта: https://github.com/EgorOrachyov/Ignimbrite