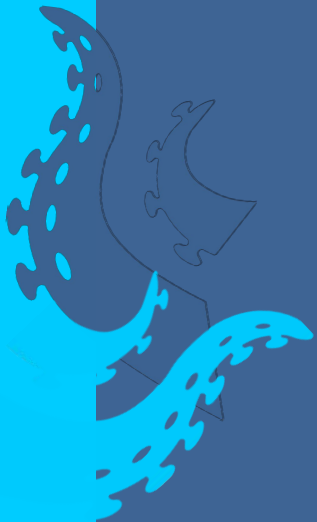
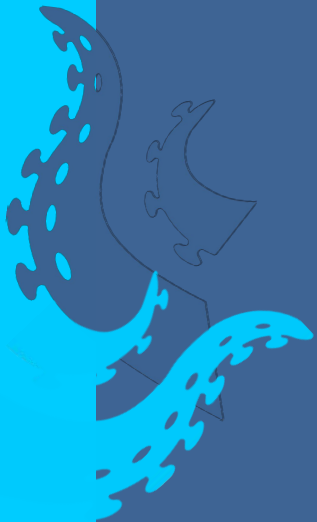


# OCTOPUS GAMES

Alexandr Murashko

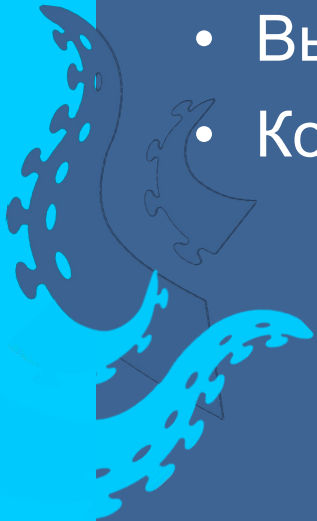


# Ускоряем игру на UE4: инструкция разработчика



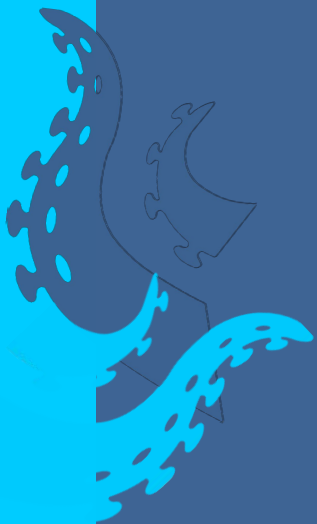
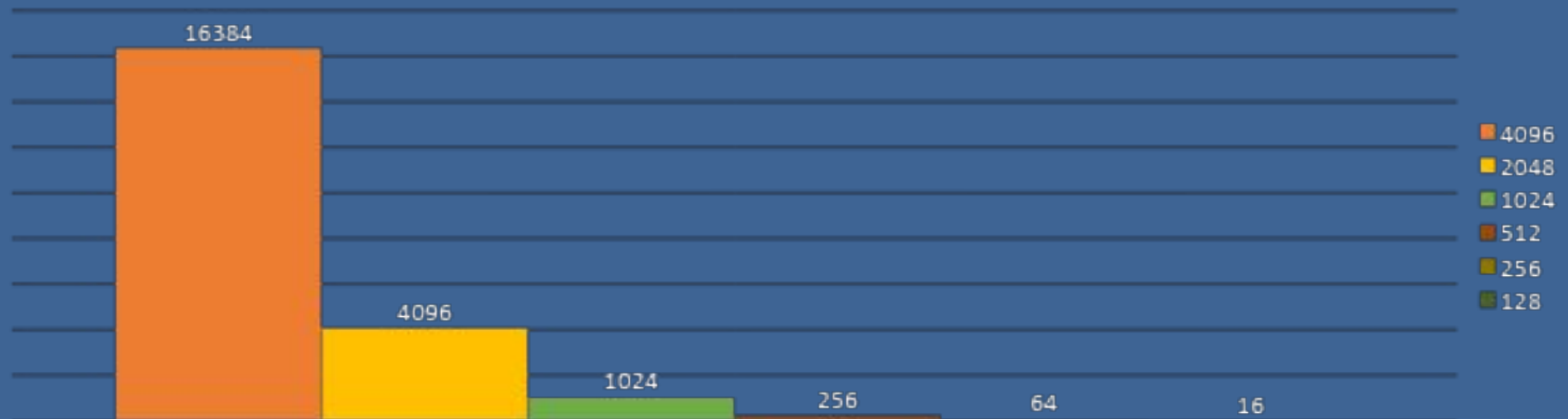
# Оптимизация текстур

- Уменьшение размера
- Сжатие
- «Девятка»
- Выравнивание размера по степени 2
- Комбинирование с помощью пиксельных шейдеров

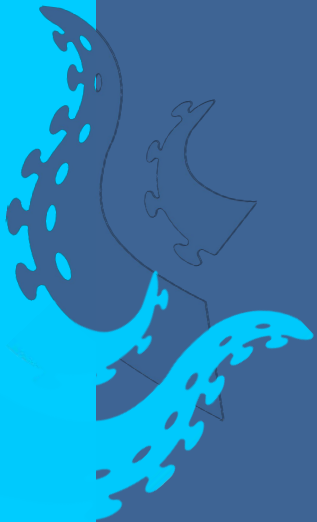
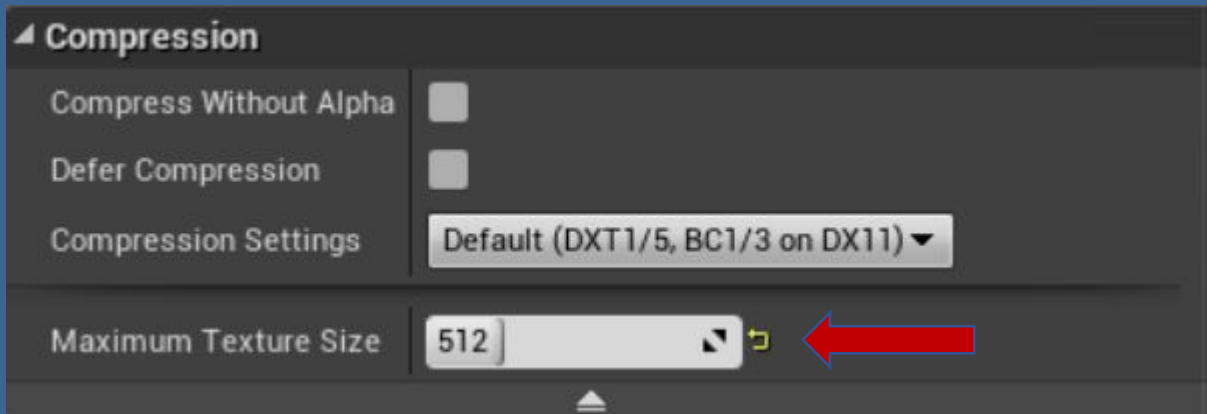


# Оптимизация текстур: уменьшение размера

Размер в килобайтах одного канала текстуры



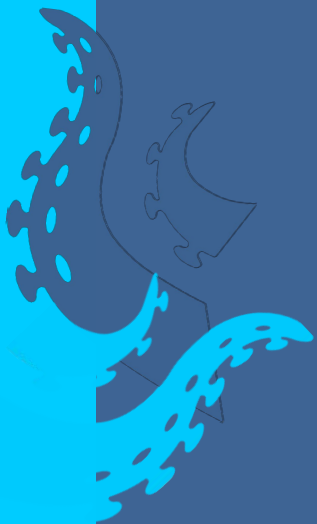
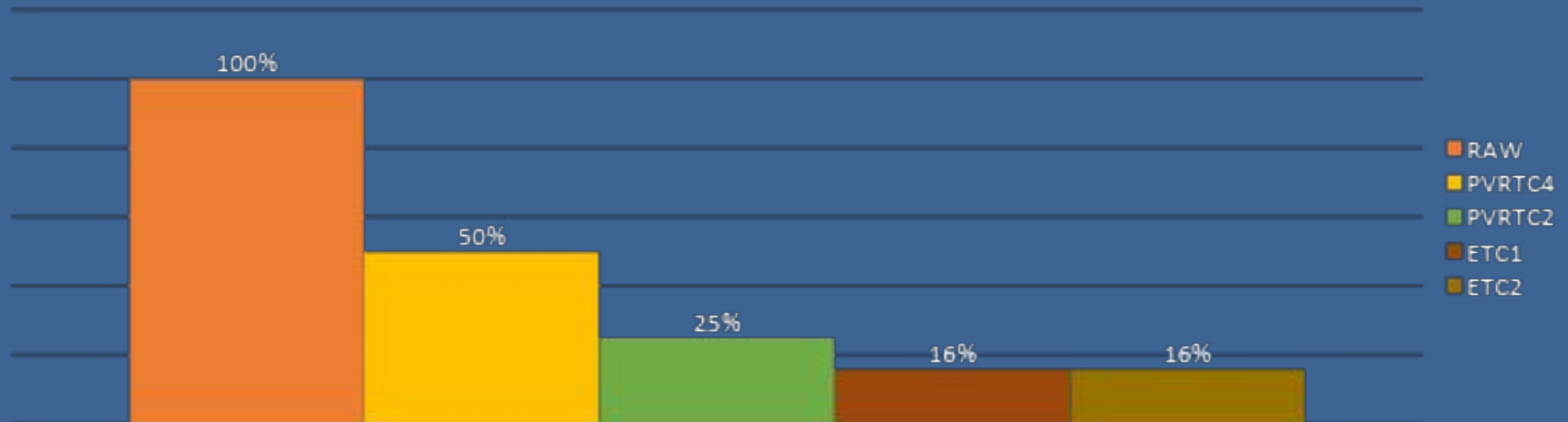
# Оптимизация текстур: уменьшение размера



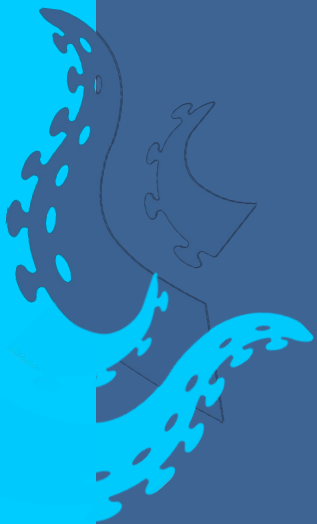
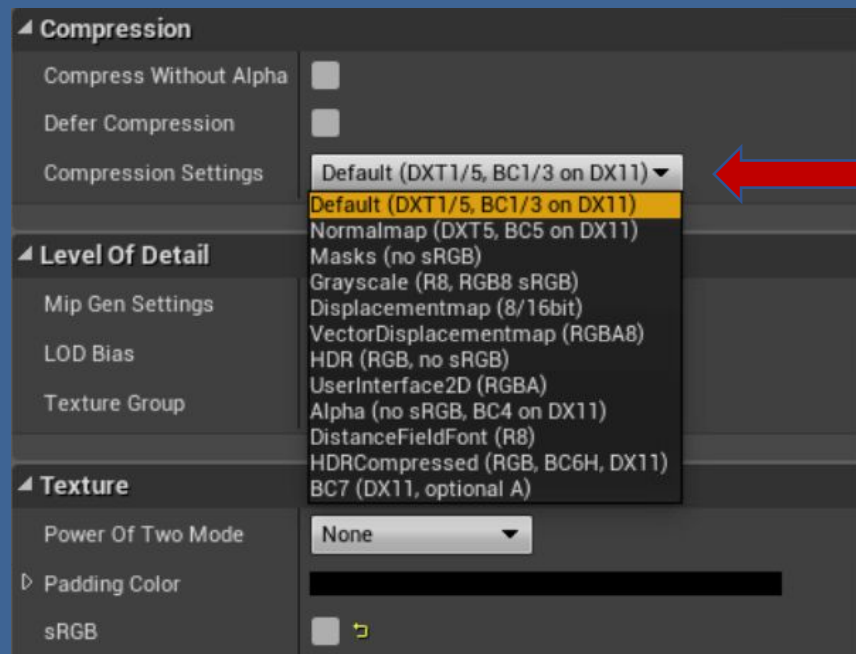
# Оптимизация текстур: сжатие

Соотношение размера текстуры от формата сжатия

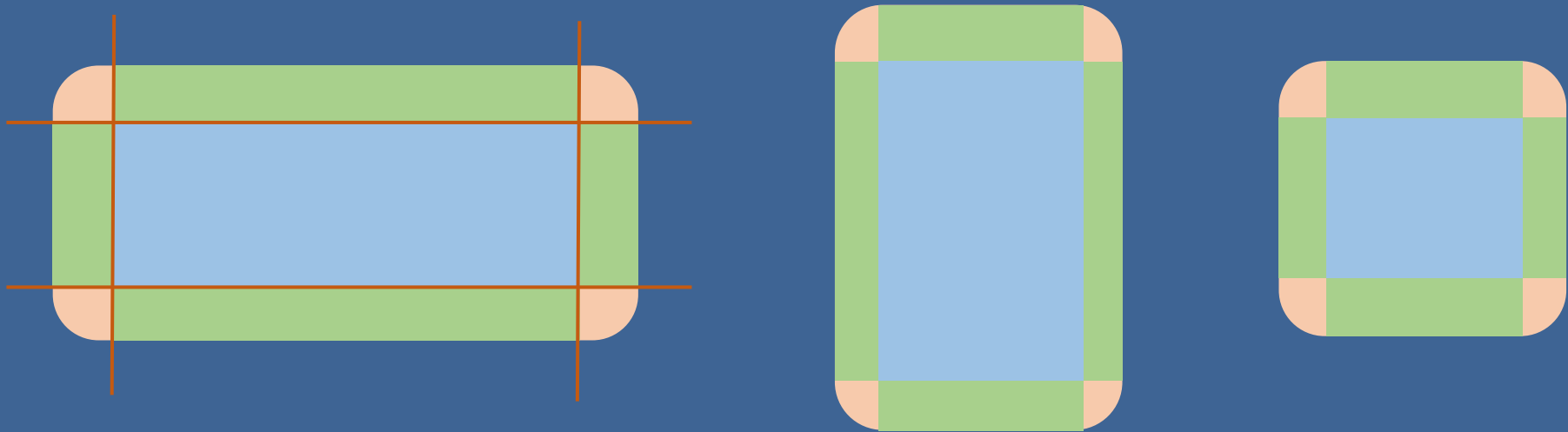
оригинала



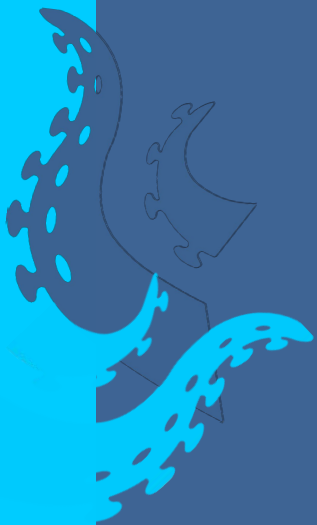
# Оптимизация текстур: сжатие



# Оптимизация текстур: «девятка»

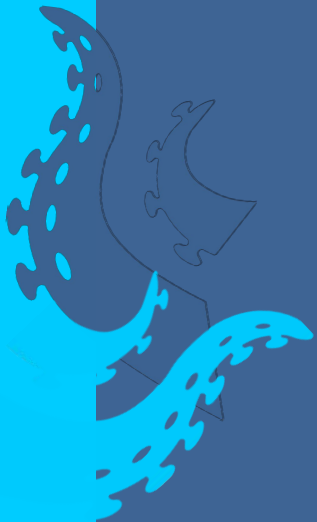
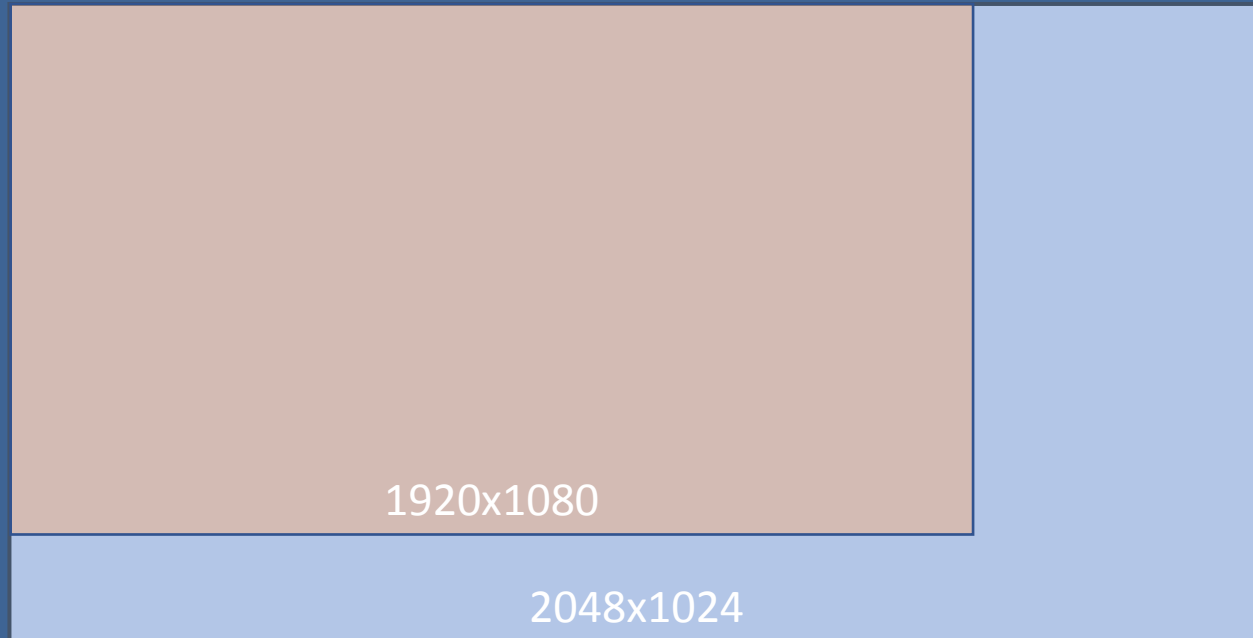


Углы фиксированы, центр и края растягиваются

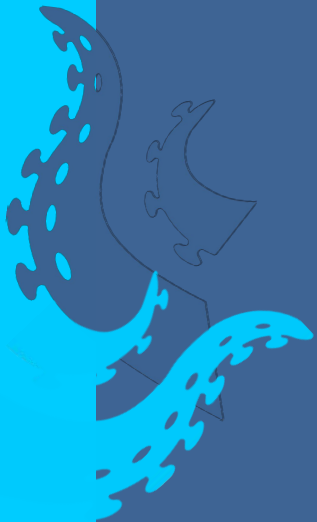
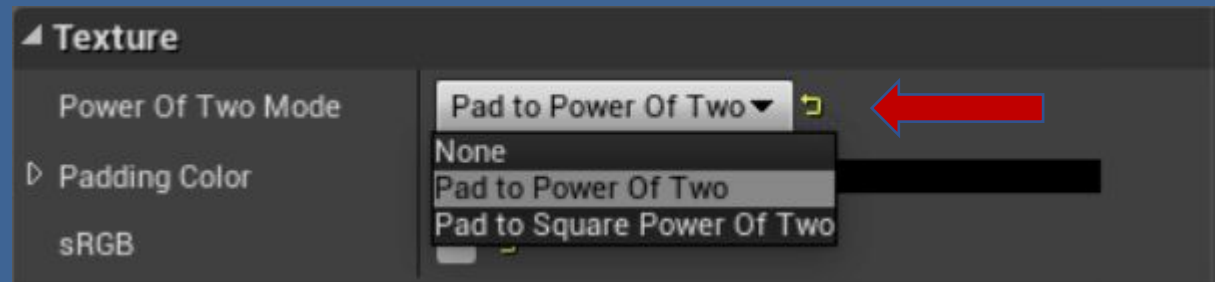




# Оптимизация текстур: выравнивание размера по степени 2



# Оптимизация текстур: выравнивание размера по степени 2



# Оптимизация текстур: комбинирование с помощью пиксельных шейдеров

Y: 1024x512  
512 Кбайт

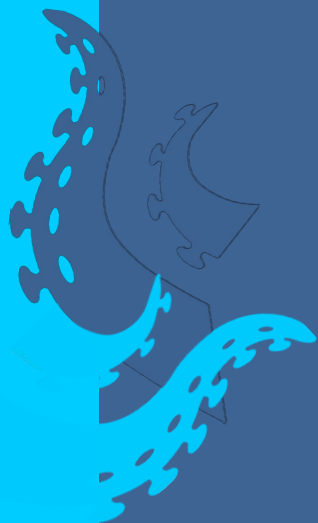
U: 512x256  
128 Кбайт

V: 512x256  
128 Кбайт

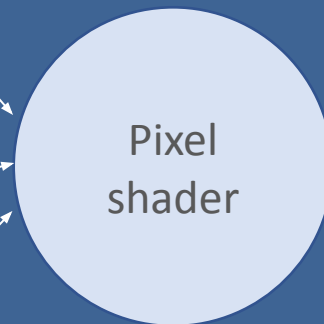
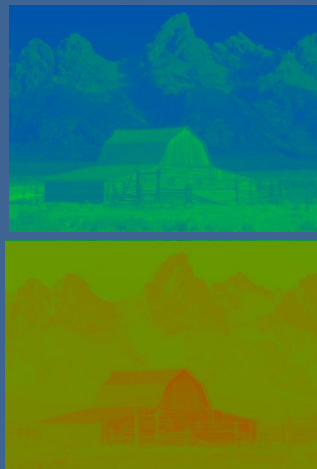
Pixel  
shader

RGB: 1024x512  
768 Кбайт

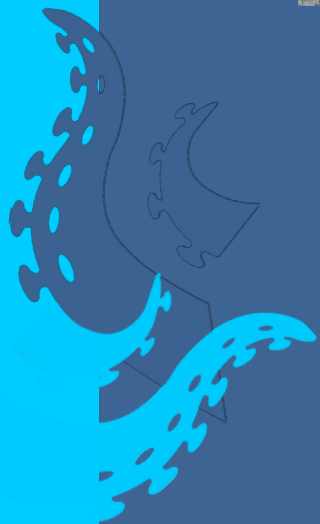
Оригинальный размер 1536 Кбайт  
Выигрыш в размере 50%



# Оптимизация текстур: комбинирование с помощью пиксельных шейдеров

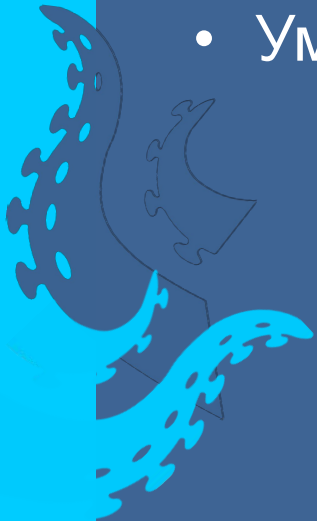


YUV444 в RGB888

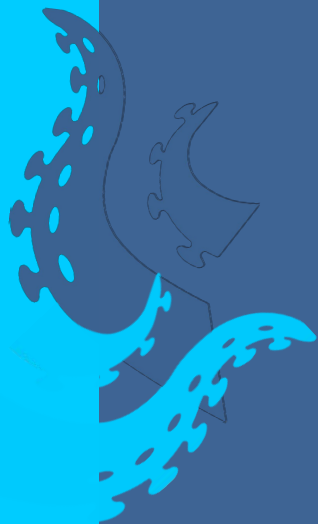
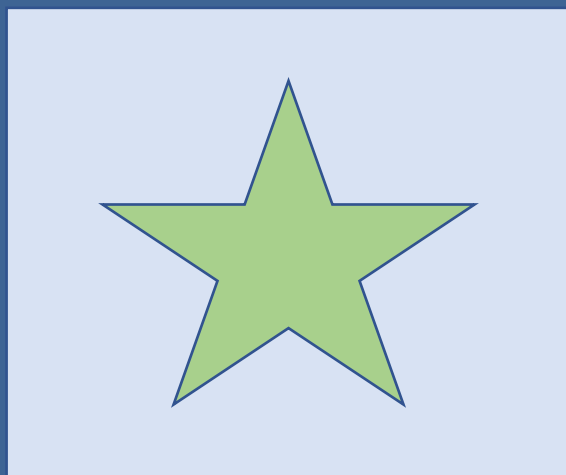


# Оптимизация GPU

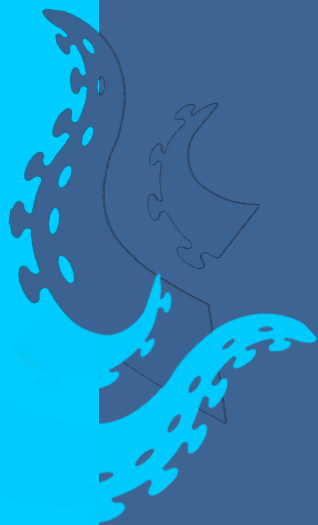
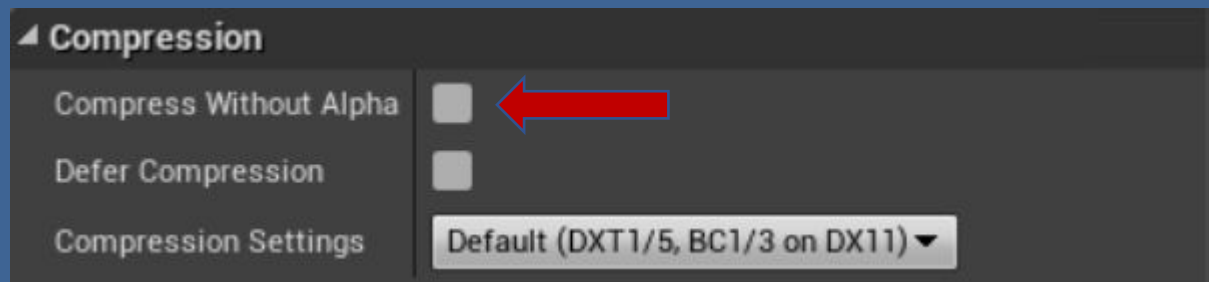
- Исключаем текстуры с альфа-каналом
- MIP-map текстуры
- Детализация модели
- Уменьшение разрешения Viewport



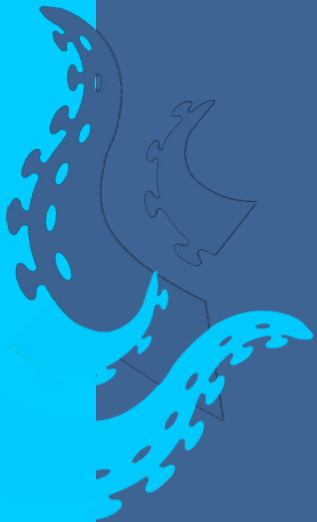
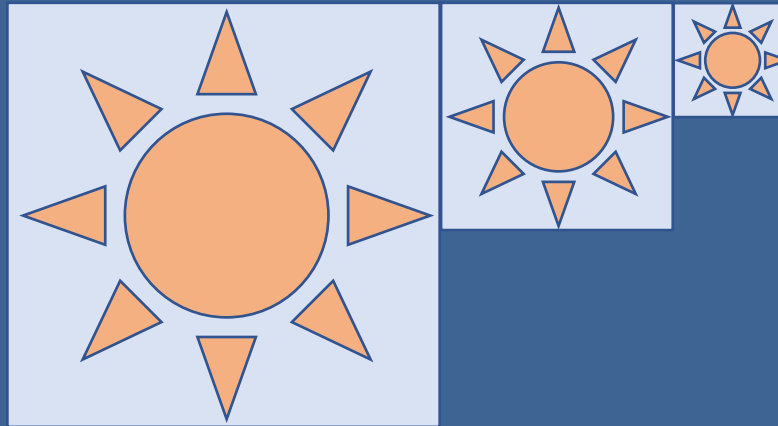
# Оптимизация GPU: исключаем текстуры с альфа-каналом



# Оптимизация GPU: исключаем текстуры с альфа-каналом

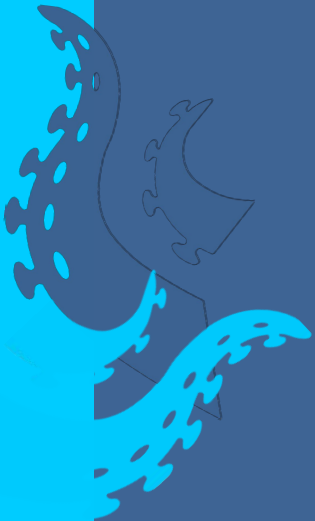
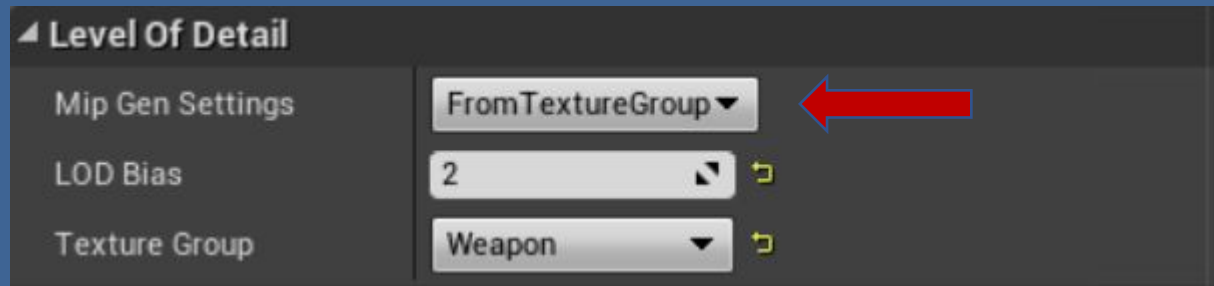


# Оптимизация GPU: MIP-map текстуры

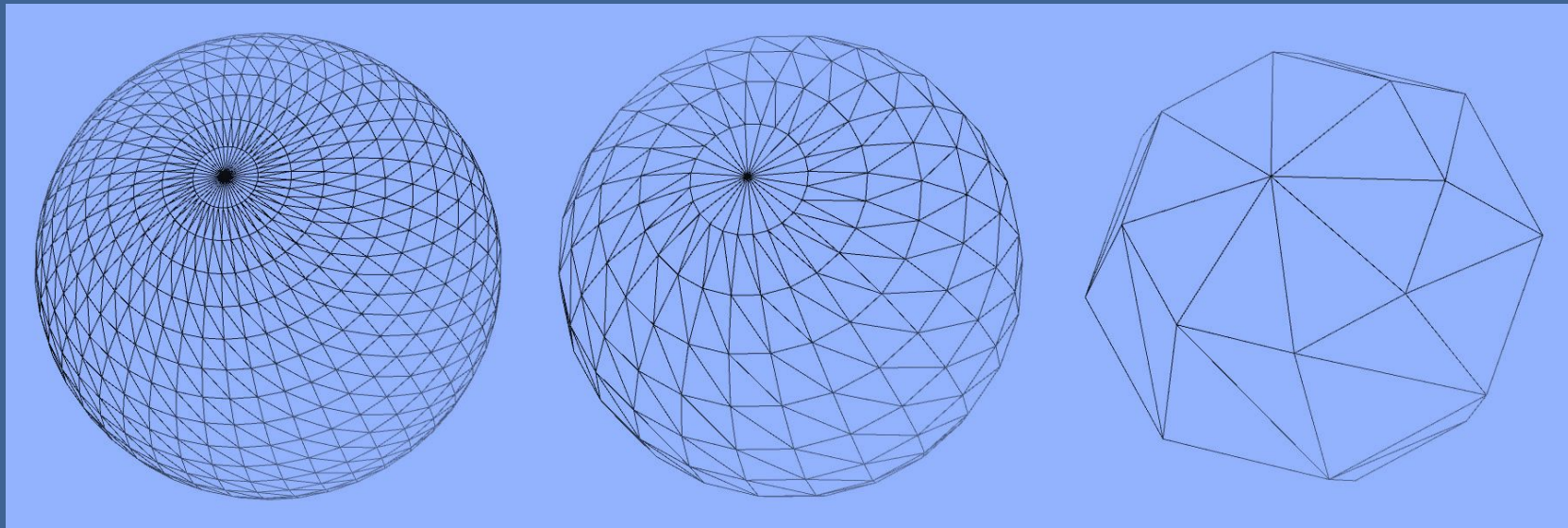




# Оптимизация GPU: MIP-map текстуры



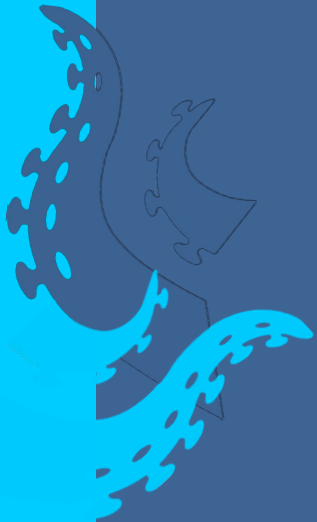
# Оптимизация GPU: детализация модели



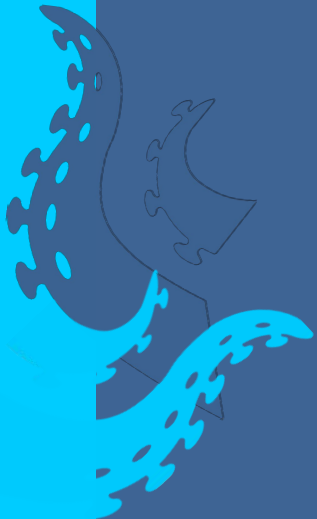
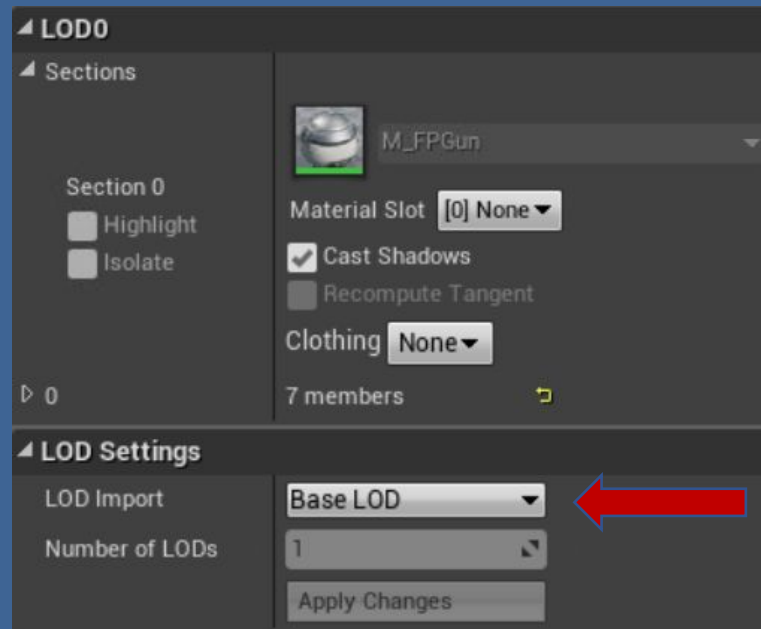
5500

1580

140

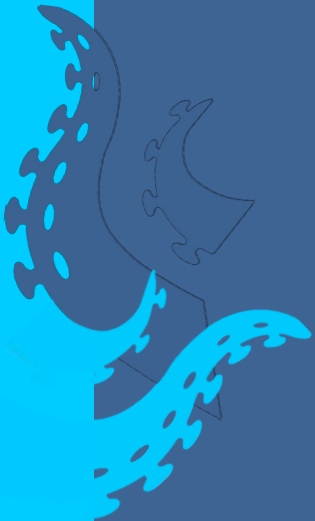


# Оптимизация GPU: детализация модели



# Оптимизация GPU: уменьшение разрешения Viewport

- `r.SetRes`
- `r.MobileContentScaleFactor`



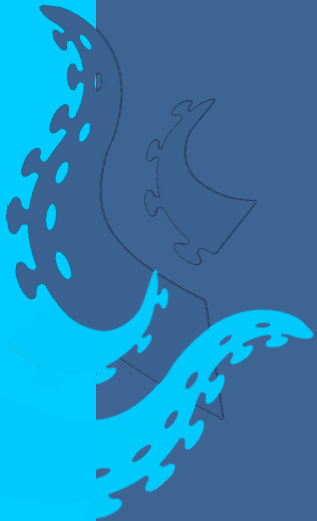
# Оптимизация CPU

- Выносим игровую логику и сложные расчеты из Blueprint в C++
- Правильно определяем Blueprint методы в C++
- Кэширование результатов расчетов
- Выносим ресурсоемкие расчеты в фоновые задачи
- Для общей оценки используем приближенные вычисления
- Правильно используем события и делегаты
- Выравниваем структуры данных в памяти
- Используем векторные инструкции процессора
- Не допускаем продолжительный нагрев процессора



# Оптимизация файловой системы

- Общие ресурсы для всех карт находятся в отдельном паке
- Специфические ресурсы для карт лучше дублировать
- Асинхронная загрузка ресурсов (streaming)



# Оптимизация файловой системы:

асинхронное прерывание

