Компьютерная графика

Практика 5: Текстуры

2021

Задание 1

Добавляем текстурные координаты

- Добавляем в вершины новое поле (например, `std::uint16_t texcoords[2];` - тип не принципиален)

Задание 1

Добавляем текстурные координаты

- Добавляем в вершины новое поле (например, `std::uint16_t texcoords[2];` - тип не принципиален)
- Добавляем значение этого поля в вершины массива `plane_vertices`

Задание 1

Добавляем текстурные координаты

- Добавляем в вершины новое поле (например, `std::uint16_t texcoords[2];` - тип не принципиален)
- Добавляем значение этого поля в вершины массива `plane_vertices`
- Описываем новый атрибут для VAO

Задание 1

Добавляем текстурные координаты

- Добавляем в вершины новое поле (например, `std::uint16_t texcoords[2];` - тип не принципиален)
- Добавляем значение этого поля в вершины массива `plane_vertices`
- Описываем новый атрибут для VAO
- Добавляем входной атрибут (`vec2`) в вершинном шейдере, и передаём его во фрагментный

Задание 1

Добавляем текстурные координаты

- Добавляем в вершины новое поле (например, `std::uint16_t texcoords[2];` - тип не принципиален)
- Добавляем значение этого поля в вершины массива `plane_vertices`
- Описываем новый атрибут для VAO
- Добавляем входной атрибут (`vec2`) в вершинном шейдере, и передаём его во фрагментный
- Во фрагментном шейдере выводим текстурные координаты в качестве цвета (например, `out_color = vec4(texcoord, 0.0, 1.0)`)

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`
- Выставляем для него min и mag фильтры в `GL_NEAREST`

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`
- Выставляем для него min и mag фильтры в `GL_NEAREST`
- В качестве данных загружаем один пиксель (например, красного цвета) в формате `GL_RGBA8` (пиксель может быть `std::uint8_t[4]` или `std::uint32_t`)

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`
- Выставляем для него min и mag фильтры в `GL_NEAREST`
- В качестве данных загружаем один пиксель (например, красного цвета) в формате `GL_RGBA8` (пиксель может быть `std::uint8_t[4]` или `std::uint32_t`)
- Во фрагментном шейдере добавляем uniform-переменную типа `sampler2D` и берём из неё цвет

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`
- Выставляем для него min и mag фильтры в `GL_NEAREST`
- В качестве данных загружаем один пиксель (например, красного цвета) в формате `GL_RGBA8` (пиксель может быть `std::uint8_t[4]` или `std::uint32_t`)
- Во фрагментном шейдере добавляем uniform-переменную типа `sampler2D` и берём из неё цвет
- В значение uniform-переменной записываем 0 (`glUniform1i`)

Задание 2

Создаём и рисуем текстуру в один пиксель

- Создаём объект текстуры типа `GL_TEXTURE_2D`
- Выставляем для него min и mag фильтры в `GL_NEAREST`
- В качестве данных загружаем один пиксель (например, красного цвета) в формате `GL_RGBA8` (пиксель может быть `std::uint8_t[4]` или `std::uint32_t`)
- Во фрагментном шейдере добавляем uniform-переменную типа `sampler2D` и берём из неё цвет
- В значение uniform-переменной записываем 0 (`glUniform1i`)
- Перед рисованием делаем нашу текстуру текущей для texture unit'а номер 0 (`glActiveTexture` + `glBindTexture`)

Задание 3

Меняем картинку на шахматную доску

- Выбираем размеры картинки (например, `width = height = 256`)

Задание 3

Меняем картинку на шахматную доску

- Выбираем размеры картинки (например, `width = height = 256`)
- Создаём массив пикселей (лучше всего - `std::vector<std::uint32_t>` или `std::vector<std::uint8_t>`)

Задание 3

Меняем картинку на шахматную доску

- Выбираем размеры картинки (например, `width = height = 256`)
- Создаём массив пикселей (лучше всего - `std::vector<std::uint32_t>` или `std::vector<std::uint8_t>`)
- В цикле заполняем пиксели чёрным или белым цветом в зависимости от `(x + y) % 2`

Задание 3

Меняем картинку на шахматную доску

- Выбираем размеры картинки (например, `width = height = 256`)
- Создаём массив пикселей (лучше всего - `std::vector<std::uint32_t>` или `std::vector<std::uint8_t>`)
- В цикле заполняем пиксели чёрным или белым цветом в зависимости от `(x + y) % 2`
- Записываем эти пиксели в текстуру (`glTexImage2D`)

Задание 4

Добавляем mipmaps

- Меняем mag filter на `GL_LINEAR`

Задание 4

Добавляем mipmaps

- Меняем mag filter на `GL_LINEAR`
- Меняем min filter на `GL_LINEAR_MIPMAP_NEAREST`

Задание 4

Добавляем mipmaps

- Меняем mag filter на `GL_LINEAR`
- Меняем min filter на `GL_LINEAR_MIPMAP_NEAREST`
- Мы не загружали mipmaps, поэтому картинка сейчас будет чёрной

Задание 4

Добавляем mipmaps

- Меняем mag filter на `GL_LINEAR`
- Меняем min filter на `GL_LINEAR_MIPMAP_NEAREST`
- Мы не загружали mipmaps, поэтому картинка сейчас будет чёрной
- Делаем `glGenerateMipmap` после загрузки основной картинки

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней
- После `glGenerateMipmap` загружаем для 1, 2 и 3 уровней монотонные красную, синюю и зелёную картинки соответственно

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней
- После `glGenerateMipmap` загружаем для 1, 2 и 3 уровней монотонные красную, синюю и зелёную картинки соответственно
- N.B.: какой уровень загрузить - второй параметр `glTexImage2D`

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней
- После `glGenerateMipmap` загружаем для 1, 2 и 3 уровней монотонные красную, синюю и зелёную картинки соответственно
- N.B.: какой уровень загрузить - второй параметр `glTexImage2D`
- N.B.: если картинка 1024x1024, первый mipmap должен быть 512x512, второй - 256x256, и т.д.

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней
- После `glGenerateMipmap` загружаем для 1, 2 и 3 уровней монотонные красную, синюю и зелёную картинки соответственно
- N.B.: какой уровень загрузить - второй параметр `glTexImage2D`
- N.B.: если картинка 1024x1024, первый mipmap должен быть 512x512, второй - 256x256, и т.д.
- N.B.: массив красных пикселей в формате `uint32_t` можно создать как

```
std::vector<std::uint32_t> pixels(512 * 512,
    0xff0000ffu);
```

Задание 5

Делаем mipmaps явно видимыми

- Можно увеличить размер картинки до 1024x1024, чтобы влезло больше mipmap-уровеней
- После `glGenerateMipmap` загружаем для 1, 2 и 3 уровней монотонные красную, синюю и зелёную картинки соответственно
- N.B.: какой уровень загрузить - второй параметр `glTexImage2D`
- N.B.: если картинка 1024x1024, первый mipmap должен быть 512x512, второй - 256x256, и т.д.
- N.B.: массив красных пикселей в формате `uint32_t` можно создать как

```
std::vector<std::uint32_t> pixels(512 * 512,
    0xff0000ffu);
```

- Сейчас mipmap-уровни видны по отдельности - меняем min filter на `GL_LINEAR_MIPMAP_LINEAR`

Задание 6

Добавляем вторую текстуру

- Повторяем всё из задания 2: создаём и настраиваем текстуру, создаём uniform-переменную

Задание 6

Добавляем вторую текстуру

- Повторяем всё из задания 2: создаём и настраиваем текстуру, создаём uniform-переменную
- В созданную uniform-переменную (`sampler2D`) записываем 1 (`glUniform1i`) - это значит, что соответствующая текстура будет взята из texture unit'а с номером 1

Задание 6

Добавляем вторую текстуру

- Повторяем всё из задания 2: создаём и настраиваем текстуру, создаём uniform-переменную
- В созданную uniform-переменную (`sampler2D`) записываем 1 (`glUniform1i`) - это значит, что соответствующая текстура будет взята из texture unit'а с номером 1
- Перед рисованием делаем старую текстуру текущей для texture unit'а номер 0, а новую - для texture unit'а номер 1 (`glActiveTexture` + `glBindTexture`, два раза)

Задание 6

Добавляем вторую текстуру

- Повторяем всё из задания 2: создаём и настраиваем текстуру, создаём uniform-переменную
- В созданную uniform-переменную (`sampler2D`) записываем 1 (`glUniform1i`) - это значит, что соответствующая текстура будет взята из texture unit'а с номером 1
- Перед рисованием делаем старую текстуру текущей для texture unit'а номер 0, а новую - для texture unit'а номер 1 (`glActiveTexture` + `glBindTexture`, два раза)
- Загружаем в текстуру данные из `test_image`, шириной `test_image_width`, высотой `test_image_height`, `format = GL_RGB`, `type = GL_UNSIGNED_BYTE`

Задание 6

Добавляем вторую текстуру

- Повторяем всё из задания 2: создаём и настраиваем текстуру, создаём uniform-переменную
- В созданную uniform-переменную (`sampler2D`) записываем 1 (`glUniform1i`) - это значит, что соответствующая текстура будет взята из texture unit'а с номером 1
- Перед рисованием делаем старую текстуру текущей для texture unit'а номер 0, а новую - для texture unit'а номер 1 (`glActiveTexture` + `glBindTexture`, два раза)
- Загружаем в текстуру данные из `test_image`, шириной `test_image_width`, высотой `test_image_height`, `format = GL_RGB`, `type = GL_UNSIGNED_BYTE`
- Во фрагментном шейдере читаем из обеих текстур и в качестве цвета выводим среднее значение