

Компьютерная графика

Практика 7: Освещение

2025



Задание 1

Добавляем ambient освещение

- Во фрагментном шейдере заводим две новые uniform-переменные: `vec3 albedo` и `vec3 ambient_light`
- Перед рисованием устанавливаем их значения:
 - Ambient light – что-нибудь маленькое, например `(0.2, 0.2, 0.2)`
 - Albedo – цвет объекта, выберите сами
- Во фрагментном шейдере делаем
`color = albedo * ambient_light`



Задание 2

Добавляем диффузное освещение от направленного источника

- Во фрагментном шейдере заводим две новые uniform-переменные: `vec3 sun_direction` и `vec3 sun_color`
- Перед рисованием устанавливаем их значения:
 - Sun direction – какой-нибудь единичный вектор, направленный примерно вверх и вперёд
 - Sun color – цвет солнца, например `(1.0, 0.9, 0.8)`

- Во фрагментном шейдере заводим функцию `vec3 diffuse(vec3 direction)`, которая вычисляет силу диффузного освещения от источника света в данном направлении:

```
return albedo * max(0.0, dot(normal, direction));
```

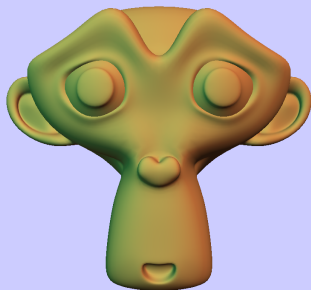
- Используем её для вычисления освещения `diffuse(sun_direction) * sun_color` и прибавляем это к результирующему цвету



Задание 3

Добавляем точечный источник света

- Во фрагментном шейдере заводим три новые uniform-переменные: `vec3 point_light_position`, `vec3 point_light_color` и `vec3 point_light_attenuation`
- Перед рисованием устанавливаем их значения:
 - Position – позиция, меняющаяся со временем
 - Color – цвет источника, выберите сами
 - Attenuation – коэффициенты затухания C_0, C_1, C_2 , например `(1.0, 0.0, 0.01)`
- Во фрагментном шейдере находим расстояние до источника света и вектор направления на него
- Используем функцию `diffuse` для вычисления освещения аналогично солнцу, домножаем на коэффициент затухания и прибавляем это к результирующему цвету
- Итого к этому моменту `color` будет суммой трёх составляющих: `ambient`, солнце и точечный источник



Задание 4

Добавляем specular освещение

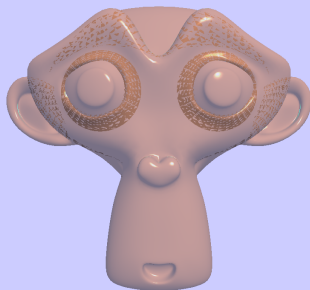
- Во фрагментном шейдере заводим две новые uniform-переменные: `float glossiness` и `float roughness`
- Перед рисованием устанавливаем их значения:
 - Glossiness – сила отражения, например `5.0`
 - Roughness – ‘шершавость’ поверхности, например `0.1` (гладкая поверхность)
- Во фрагментном шейдере добавляем функцию `vec3 specular(vec3 direction)`, которая вычисляет specular составляющую (см. слайды лекции)

```
return glossiness * albedo * pow(max(0.0,  
    dot(reflected, view_direction)), power);
```
- Прибавляем specular составляющую к вкладам обоих источников света, т.е. заменяем `diffuse(direction)` на `diffuse(direction) + specular(direction)`



Играем с прозрачностью

- Если переменная `transparent` (она уже есть в коде, значение меняется по пробелу) имеет значение `true`, включаем прозрачность:
 - `glEnable(GL_BLEND)`
 - `glBlendEquation`
 - `glBlendFunc`
- Иначе выключаем прозрачность `glDisable(GL_BLEND)`
- Не забудем записать в альфа-канал во фрагментном шейдере значение, отличное от 1 (например, 0.5)



Задание 6*

Рисуем несколько моделей с разными параметрами материала

- Рисуем модель девять раз квадратом 3x3 (в плоскости XY)
- У моделей должны отличаться параметры glossiness и roughness:
 - Roughness принимает значения примерно от 0.1 до 0.5
 - Значения glossiness подберите сами

