

A. Apa itu Lighting dan Shadow?

Cahaya adalah bentuk radiasi elektromagnetik yang terlihat oleh mata manusia, membantu kita dalam memahami lingkungan sekitar. Dengan bantuan cahaya, mata dapat menangkap bentuk dan warna objek di sekitar kita.

Dalam konteks komputer grafis, konsep pencahayaan menjadi kunci dalam menciptakan visual yang realistis. Pengaturan cahaya dalam sebuah render memengaruhi penampilan objek secara signifikan. Ada berbagai jenis pencahayaan yang umum digunakan, seperti pencahayaan ambient yang memberikan cahaya merata ke seluruh objek, serta pencahayaan titik atau directional yang memberikan sumber cahaya yang terfokus dan menghasilkan bayangan yang jelas. Selain itu, konsep pencahayaan dalam rendering komputer juga melibatkan pemodelan material.

Sedangkan bayangan adalah akibat dari tidak adanya cahaya akibat oklusi. Bila berkas cahaya suatu sumber cahaya tidak mengenai suatu benda karena terhalang oleh benda lain, maka benda tersebut berada dalam bayangan. Bayangan menambahkan banyak realisme pada pemandangan yang terang dan memudahkan pemirsa mengamati hubungan spasial antar objek. Mereka memberikan kesan kedalaman yang lebih besar pada pemandangan dan objek kita.

B. Mengapa Mengimplementasi/Mensimulasikan Lighting & Shadow itu Penting?

Pencahayaan memiliki peran sentral dalam dunia desain visual dan komputer grafis. Hal ini bukan hanya tentang menambahkan cahaya ke suatu gambar, tetapi tentang menciptakan suasana, menyoroti detail, dan memperkuat pesan yang ingin disampaikan. Dengan mengatur intensitas, warna, dan arah cahaya, seorang desainer dapat mengubah sepenuhnya cara kita melihat dan merasakan sebuah gambar. Misalnya, pencahayaan yang lembut dan merata dapat menciptakan suasana yang tenang dan damai, sementara pencahayaan yang kontras dan dramatis dapat menimbulkan perasaan tegang atau misterius. Pencahayaan juga memainkan peran penting dalam menyoroti detail dan menciptakan hierarki visual, membantu pengamat untuk fokus pada aspek-aspek yang paling penting dari sebuah desain.

C. Faktor-Faktor yang Memengaruhi Lighting

1. **Posisi dan Arah Datang Cahaya:** Posisi dan arah datang cahaya berpengaruh pada distribusi cahaya dalam suatu scene. Cahaya yang datang dari arah yang berbeda

dapat menghasilkan bayangan yang berbeda. Dalam opengl `LightPosition` digunakan untuk menetapkan posisi sumber cahaya dalam koordinat dunia.

2. **Material Objek:** Sifat material objek, seperti kemampuan untuk memantulkan cahaya dan tekstur permukaannya, juga memengaruhi tampilan pencahayaan.
3. **Warna Cahaya:** Spektrum warna yang dihasilkan oleh sumber cahaya. Ini bisa berwarna putih, kuning, biru, dan sebagainya. Warna cahaya yang berbeda akan memberikan tampilan yang berbeda pada objek.
4. **Jarak Cahaya:** Jarak antara sumber cahaya dan objek memengaruhi intensitas cahaya yang diterima oleh objek. Pengaruh jarak cahaya tercermin dalam perhitungan komponen pencahayaan difusi dan spekular dalam kode tersebut.kan permukaan yang halus cenderung memantulkan cahaya secara spekular.
5. **Tipe Pencahayaan (light caster):** Jenis pencahayaan yang digunakan, seperti Point Light, directional light, spotlight juga memengaruhi tampilan akhir suatu objek. Setiap jenis pencahayaan memiliki karakteristik unik yang mempengaruhi bagaimana cahaya tersebar dalam objek.

D. Implementasi

Untuk implementasi lighting dalam grafika komputer, kita memerlukan sebuah shader. Shader adalah program kecil yang dijalankan pada unit pemrosesan grafis (GPU) untuk mengontrol bagaimana piksel (fragment shader) atau titik (vertex shader) dalam scene dirender. Shader memiliki dua komponen utama:

- a. **Vertex Shader:** Vertex shader bertanggung jawab untuk memanipulasi properti-properti setiap titik dalam objek 3D sebelum ditampilkan di layar. Komponen dari vertex shader biasanya mencakup:
 - Transformasi: Mengubah koordinat titik objek ke koordinat layar menggunakan matriks transformasi.
 - Penerangan: Menghitung pengaruh pencahayaan terhadap setiap titik berdasarkan posisi, arah, dan sifat-sifat cahaya serta material objek.
 - Interpolasi: Menghasilkan data interpolasi, seperti warna atau koordinat tekstur, untuk setiap titik yang akan digunakan oleh fragment shader.
- b. **Fragment Shader:** Fragment shader bertanggung jawab untuk menentukan warna akhir (dan beberapa properti lainnya) dari setiap piksel dalam gambar yang dihasilkan. Komponen dari fragment shader mencakup:

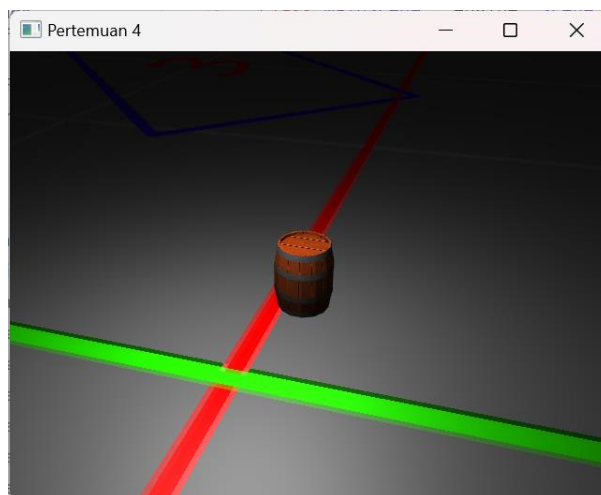
- Penerangan: Menggunakan informasi pencahayaan dari vertex shader dan sifat material untuk menghitung warna akhir piksel.
- Tekstur: Memetakan warna piksel berdasarkan koordinat tekstur dan gambar tekstur yang terkait.
- Efek visual: Menerapkan efek visual tambahan, seperti bayangan atau efek partikel.

Dengan demikian, shader merupakan komponen kunci dalam menciptakan efek pencahayaan yang realistis dan visualisasi yang menarik dalam aplikasi grafis 3D. Untuk contoh implementasinya sendiri dapat dilihat berdasarkan beberapa kategori faktor yang mempengaruhi lighting.

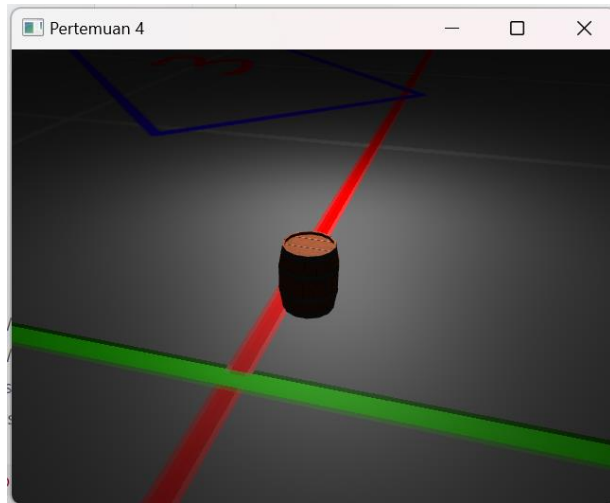
1. Posisi dan Arah Datang Cahaya

Posisi cahaya, yang ditentukan oleh `'LightPosition'`, menentukan lokasi titik cahaya dalam ruang 3D dengan koordinat x, y, dan z. Dengan menentukan nilai x, y, dan z pada `'LightPosition'`, kita dapat menentukan di mana titik cahaya berada dalam scene.

Contoh visualisasi dari pengaturan `'LightPosition'` adalah sebagai berikut:



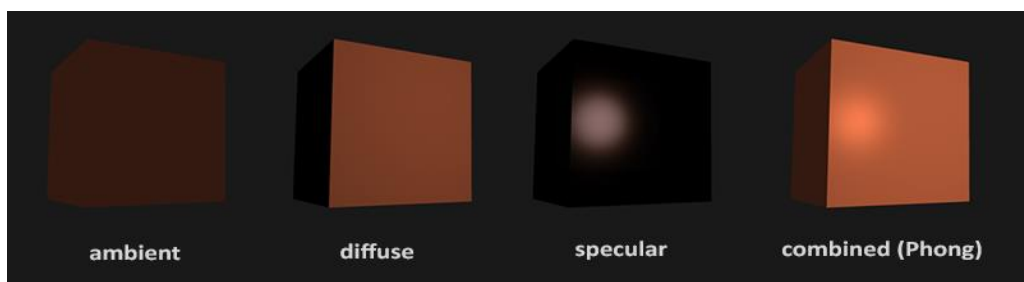
Dalam contoh pertama, dengan `'LightPosition' = vec3(0, 10, 0)` dan `'posisi kamera' = vec3(-10, 10, 5)`, maka cahaya akan terlihat pada sisi sebelah depan kanan dari barrel. Ini terjadi karena titik cahaya berada di atas dan sedikit ke depan dari posisi kamera, menyebabkan bayangan dan refleksi cahaya terpantul ke arah kanan.



Namun, jika `'LightPosition'` diubah menjadi `'vec3(10, 10, 0)'` dengan posisi kamera yang sama, maka posisi cahaya juga akan berubah. Akibatnya, cahaya akan terlihat berpusat lebih ke belakang dari posisi kamera dan membelakangi barel. Perubahan ini akan mempengaruhi cara cahaya jatuh pada objek, yang mungkin menghasilkan bayangan dan refleksi yang berbeda.

2. Material Objek

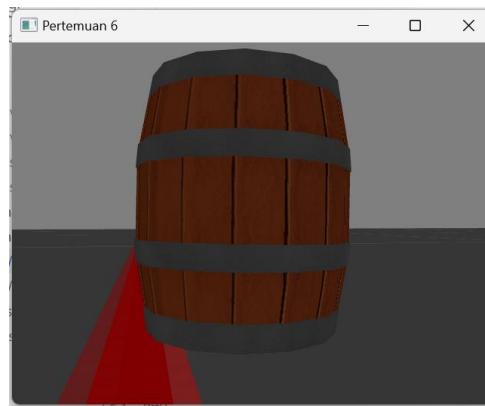
Seperti yang diketahui, ketika cahaya jatuh pada suatu permukaan maka cahaya tersebut akan memantul, pantulannya sendiri dipengaruhi oleh tekstur permukaan objek. Berdasarkan hal ini, pada grafika komputer pada model pencahayaan memiliki 3 komponen. Komponen utama model pencahayaan Phong terdiri dari 3 komponen: pencahayaan ambient, diffuse, dan specular.



a) Ambient

Pencahayaan ambient adalah komponen pencahayaan yang merujuk pada cahaya yang tersebar secara merata di dalam scene, memberikan pencahayaan keseluruhan tanpa memperhatikan arah atau sumber cahaya spesifik.

Berikut adalah contoh gambar barel dengan lighting ambient:

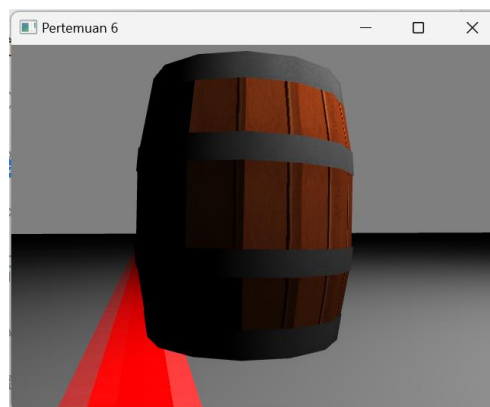


Perubahan kode ada pada shader fragment di bagian:

```
color =
    MaterialAmbientColor;
```

b) Diffuse

Pencahayaan difuse adalah komponen pencahayaan yang menggambarkan bagaimana cahaya tersebar pada permukaan objek dan tersebar secara merata ke segala arah.

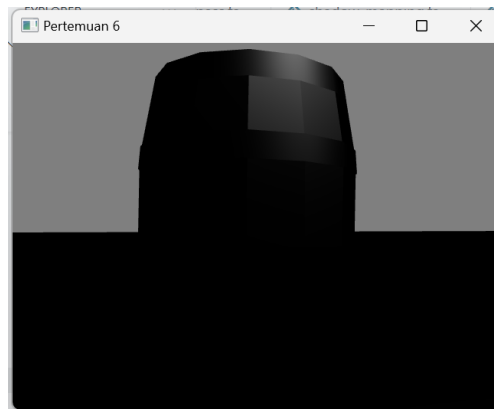


Perubahan kode ada pada shader fragment di bagian:

```
color =
    MaterialDiffuseColor * LightColor * LightPower *
    cosTheta / (distance*distance);
```

c) Specular

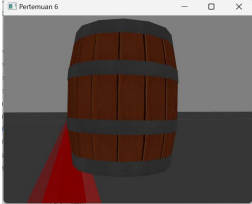
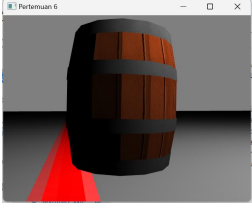
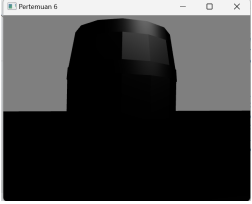


Pencahayaan specular adalah komponen pencahayaan yang menggambarkan pantulan cahaya secara langsung dari sumber cahaya, menciptakan kilauan atau highlight pada permukaan objek.



Perubahan kode ada pada shader fragment di bagian:

```
color =
    MaterialSpecularColor * LightColor * LightPower *
    pow(cosAlpha,5) / (distance*distance);
```

Perbandingan implementasi komponen lighting sebagai berikut:

Ambient	Diffuse	Specular
		
ambient + diffuse		combine
		

Berikut adalah kode lengkapnya untuk fragment shader dengan kombinasi 3 komponen:

```
#version 330 core

// Interpolated values from the vertex shaders
in vec2 UV;
in vec3 Position_worldspace;
in vec3 Normal_cameraspace;
```

```

in vec3 EyeDirection_cameraspace;
in vec3 LightDirection_cameraspace;

// Output data
out vec3 color;

// Values that stay constant for the whole mesh.
uniform sampler2D textureSampler;
uniform mat4 MV;
uniform vec3 LightPosition_worldspace;

void main(){

    // Light emission properties
    // You probably want to put them as uniforms
    vec3 LightColor = vec3(1,1,1);
    float LightPower = 160.0f;

    // Material properties
    vec3 MaterialDiffuseColor = texture( textureSampler, UV ).rgb;
    vec3 MaterialAmbientColor = vec3(0.5,0.5,0.5) *
MaterialDiffuseColor;
    vec3 MaterialSpecularColor = vec3(0.3,0.3,0.3);

    // Distance to the light
    float distance = length( LightPosition_worldspace -
Position_worldspace );

    // Normal of the computed fragment, in camera space
    vec3 n = normalize( Normal_cameraspace );

    // Direction of the light (from the fragment to the light)
    vec3 l = normalize( LightDirection_cameraspace );
    // Cosine of the angle between the normal and the light
direction,
    // clamped above 0
    // - light is at the vertical of the triangle -> 1
    // - light is perpendicular to the triangle -> 0
    // - light is behind the triangle -> 0
    float cosTheta = clamp( dot( n,l ), 0,1 );

    // Eye vector (towards the camera)
    vec3 E = normalize(EyeDirection_cameraspace);

    // Direction in which the triangle reflects the light
    vec3 R = reflect(-l,n);

```

```

    // Cosine of the angle between the Eye vector and the Reflect
vector,
    // clamped to 0
    // - Looking into the reflection -> 1
    // - Looking elsewhere -> < 1
    float cosAlpha = clamp( dot( E,R ), 0,1 );

    color =
        // Ambient : simulates indirect lighting
        MaterialAmbientColor +
        // Diffuse : "color" of the object
        MaterialDiffuseColor * LightColor * LightPower * cosTheta /
(distance*distance) + //; // +
        // Specular : reflective highlight, like a mirror
        MaterialSpecularColor * LightColor * LightPower *
pow(cosAlpha,5) / (distance*distance);
}

```

Penjelasan mengenai unsur kode diatas:

- LightColor: warna cahaya.
- LightPower: intensitas cahaya yang dipancarkan oleh sumber cahaya. Semakin tinggi nilainya, semakin kuat cahaya yang dipancarkan.
- MaterialDiffuseColor: warna difus dari material objek, yang merupakan warna dasar objek tanpa perhitungan pencahayaan. Dalam kode tersebut, warna difus berasal dari sampel tekstur pada koordinat UV objek.
- MaterialAmbientColor: warna ambien dari material objek, yang merupakan warna yang dipantulkan dari objek di sekitarnya secara merata.
- MaterialSpecularColor: warna spekular dari material objek, yang merupakan warna sorotan yang dihasilkan oleh cahaya yang dipantulkan dalam arah tertentu.
- distance: jarak antara posisi piksel dan posisi sumber cahaya dalam ruang dunia.
- n: vektor normal untuk permukaan objek pada titik tertentu, yang dinormalisasi dalam ruang kamera.
- l: vektor yang menunjukkan arah dari piksel ke sumber cahaya, yang dinormalisasi dalam ruang kamera.

- $\cos\Theta$: kosinus sudut antara vektor normal dan vektor cahaya, yang dibatasi ke dalam rentang 0 hingga 1.
- E : vektor yang menunjukkan arah pandangan dari mata ke piksel, yang dinormalisasi dalam ruang kamera.
- R : vektor yang menunjukkan arah refleksi cahaya, yang dihasilkan saat cahaya dipantulkan dari permukaan objek, dihitung dengan mengambil pantulan dari vektor cahaya terhadap vektor normal.
- $\cos\alpha$: kosinus sudut antara vektor pandangan dan vektor refleksi, yang dibatasi ke dalam rentang 0 hingga 1.
- color : hasil akhir dari perhitungan pencahayaan, yang terdiri dari komponen ambien, difus, dan spekular. Komponen-komponen ini dihitung berdasarkan sifat material objek dan posisi serta arah cahaya dan kamera.

3. Warna Cahaya

Menggunakan kode sebelumnya, kita menggunakan cahaya dengan warna putih. Nah penentuan warna cahaya ini dapat kita ubah pada `LightColor`. Sebelumnya `LightColor = vec3(1,1,1)`; sehingga menghasilkan warna putih, kali ini akan kita ubah sehingga dapat berwarna kuning.

```
LightColor = vec3(1,1,0);
```

