

SHADING

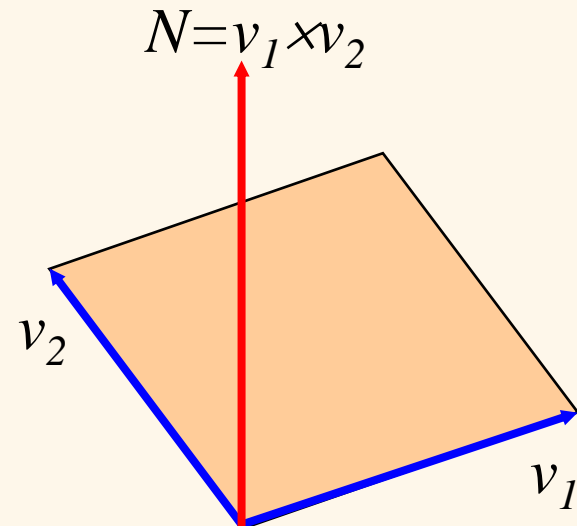
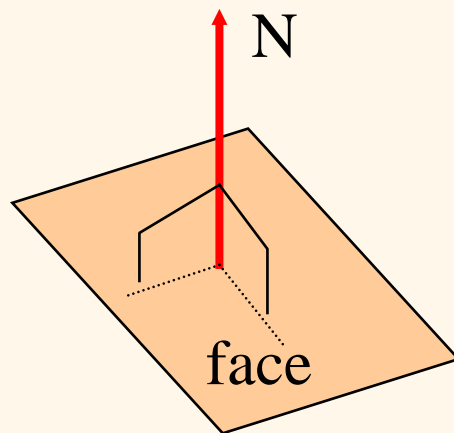
Achmad Basuki
Nana Ramadijanti

Materi

- Normal Vector
- Unit Vector
- Optical Model
- Flat Shading
- Gouraud Shading

Normal Vector

- Normal Vector adalah vector yang arahnya tegak lurus pada luasan (face)
- Normal Vector dapat diperoleh dari perkalian silang (cross-product) dari dua vector yang berada pada face
- Besar dari Normal Vector Vector tergantung pada hasil perkalian silangnya



Unit Vector

- Unit Vector adalah vektor yang besarnya adalah satu satuan dan arahnya tergantung arah vektor asalnya.
- Besar suatu vektor dapat diperoleh dengan $|v| = \sqrt{v_x^2 + v_y^2 + v_z^2}$
- Agar vektor v menjadi unit vektor maka semua koefisien (v_x, v_y, v_z) dibagi dengan $|v|$

Implementasi Unit Vector

```
vector3D_t unitVector(vector3D_t vec)
{
    int i;
    float vec2=0.;
    float vec1,invvec1;
    for(i=0;i<3;i++)
        vec2+=vec.v[i]*vec.v[i];
    vec1=sqrt(vec2);
    if (vec1!=0.) {
        invvec1=1./vec1;
        for (i=0;i<3;i++) vec.v[i]*=invvec1;
    }
    vec.v[3]=1.;
    return vec;
}
```

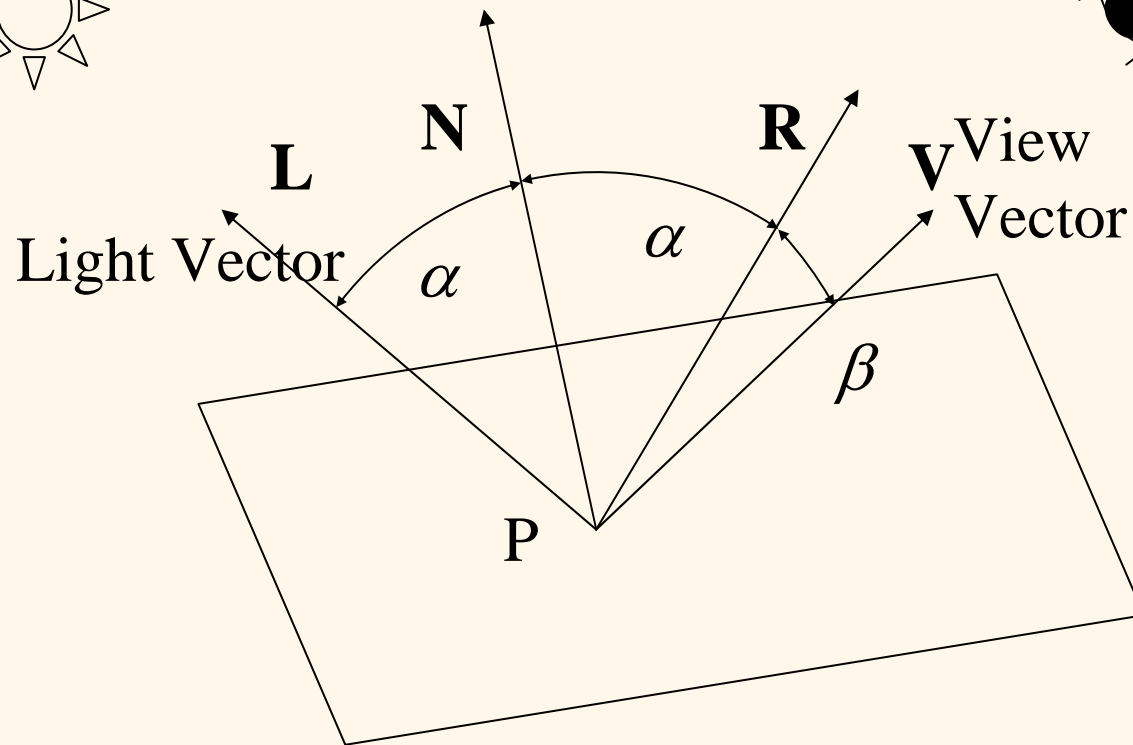
Optical Model

Illuminant



Normal Vector to
the Plane

Reflection Vector

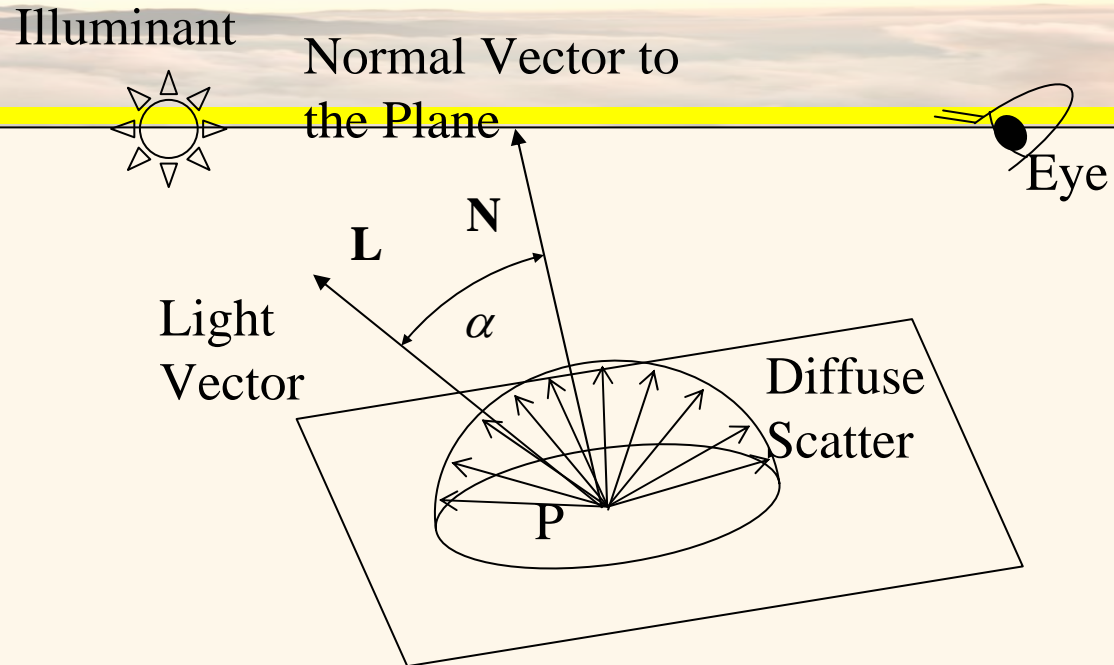


Direction Vectors $\rightarrow |\mathbf{L}| = |\mathbf{N}| = |\mathbf{R}| = |\mathbf{V}| = 1$

Macam-Macam Optical Model

- Diffuse Scattering
- Specular Reflection
- Ambient
- Lambert's Law
- **Phong Model**

Diffuse Scattering



$$I_d = I_s k_d \text{MAX}(\cos \alpha, 0)$$

$$\cos \alpha = \mathbf{L} \cdot \mathbf{N}$$

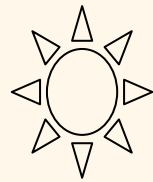
I_d : Intensity of the diffuse component

I_s : Intensity of the Light Source

k_d : Diffuse reflection coefficient

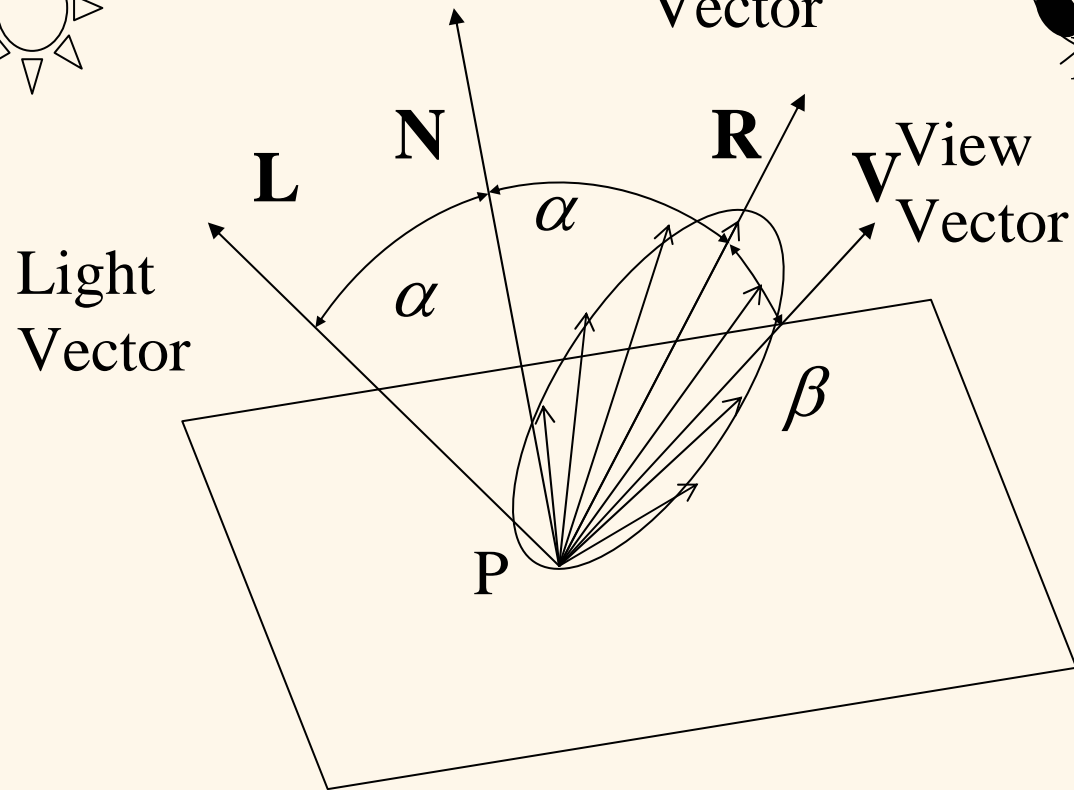
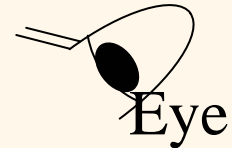
Specular Reflection

Illuminant



Normal Vector to the Plane

Reflection Vector



Specular Reflection

$$I_{sp} = I_s k_{sp} \text{MAX}(\cos^n \beta, 0)$$

$$\cos \beta = \mathbf{R} \cdot \mathbf{V}$$

$$\mathbf{R} = 2\mathbf{N}(\mathbf{L} \cdot \mathbf{N}) - \mathbf{L}$$

I_{sp} : Intensity of the Specular Reflection

I_s : Intensity of the Light Source

k_{sp} : Specular reflection coefficient

n : constant from experiment (1 - -200)

Specular Reflection

$$\mathbf{R} = 2\mathbf{N}(\mathbf{L} \cdot \mathbf{N}) - \mathbf{L}$$

$$\because \mathbf{L} \cdot \mathbf{N} = \mathbf{R} \cdot \mathbf{N}, \quad \mathbf{R} = a\mathbf{L} + b\mathbf{N}$$

$$\mathbf{L} \cdot \mathbf{N} = (a\mathbf{L} + b\mathbf{N}) \cdot \mathbf{N}$$

$$b = (1 - a)\mathbf{L} \cdot \mathbf{N}$$

$$\mathbf{R} = a\mathbf{L} + \{(1 - a)\mathbf{L} \cdot \mathbf{N}\}\mathbf{N}$$

$$\mathbf{R}^2 = 1, (\mathbf{L}^2 = 1, \mathbf{N}^2 = 1)$$

then

$$a = \pm 1 \Rightarrow a = -1$$

Ambient

$$I_a = I_s k_a$$

I_a : Intensity of the Ambient Reflection

I_s : Intensity of the Light Source

k_{sp} : Ambient reflection coefficient

Lambert Law

(Diffuse Scattering + Ambient)

$$C_L = C_s \{ k_d \text{MAX}(\cos \alpha, 0) + k_a \}$$

$$\cos \alpha = \mathbf{L} \cdot \mathbf{N}$$

C_L : Reflection Color

C_s : Surface Color

Phong Model

(Diffuse Scattering+Ambient Specular Reflection)

$$C_L = C_s \{ k_d \text{MAX}(\cos \alpha, 0) + k_a \} + C_{white} k_{sp} \text{MAX}(\cos^n \beta, 0)$$

$$\cos \alpha = \mathbf{L} \cdot \mathbf{N}$$

$$\cos \beta = \mathbf{R} \cdot \mathbf{V}$$

$$\mathbf{R} = 2\mathbf{N}(\mathbf{L} \cdot \mathbf{N}) - \mathbf{L}$$

C_L : Reflection Color

C_s : Surface Color

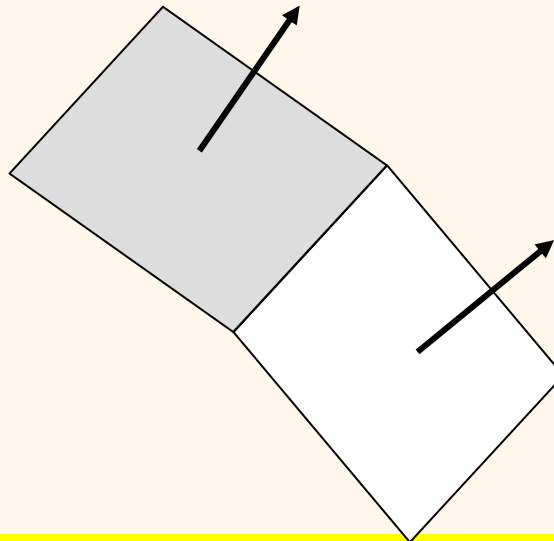
C_{white} : Specular Color (white)

Implementasi Phong Model

```
color_t PhongModel(vector3D_t Light,vector3D_t Normal,
    vector3D_t View,color_t col)
{ float kspe=0.7; // specular reflection coefficient
  float kdif=0.6; // diffuse reflection coefficient
  float kamb=0.4; // ambient light coefficient
  float tmp,NL,RV;
  color_t ColWhite={ 1,1,1 };
  vector3D_t ReflectionVector=(2.*(Light*Normal)*Normal)-Light;
  tmp=Normal*Light;
  NL=funcPositive(tmp);
  tmp=ReflectionVector*View;
  RV=funcPositive(tmp);
  return kdif*NL*col+kspe*power(RV,4)*ColWhite+kamb*col;
}
```

Flat Shading

- Satu face mempunyai warna yang sama
- Flat shading menggunakan model Phong untuk optical view



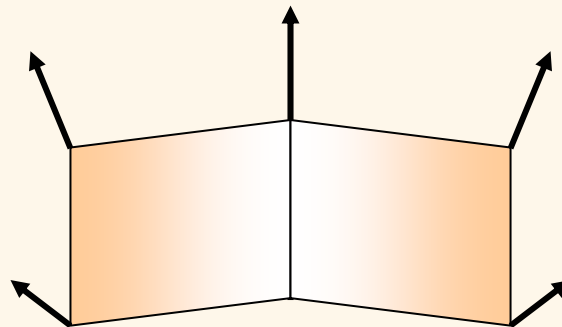
Implementasi Flat Shading

```
if(NormalVector.v[2]>0.)
{
    NormalVector=unitVector(NormalVector);
    fillcol=PhongModel(LightVector,NormalVector,ViewVector,col);
    fillPolygon(p,bola.fc[i].NumberofVertices,fillcol);
}
```

col adalah variabel warna, yang menyatakan warna dasar dari obyek

Gouraud Shading

- Satu face mempunyai gradiasi warna yang memperhitungkan tingkat kecerahan pada setiap titik pada face tersebut
- Dengan Shading ini dihasilkan warna yang halus gradiasinya
- Shading ini memerlukan vektor normal pada setiap titiknya
- Menggunakan fungsi `gradatePolygon` untuk menyatakan Gouraud Shading



Implementasi Gouraud Shading

```
for (i=0;i<sphere.NumberofVertices;i++) {  
    vec[i]=Point2Vector(sphere.pnt[i]);  
    vec[i]=mat*vec[i];  
    nrmvec[i]=mat*sphere.NormalVector[i]-ZeroVector;  
}
```

Pada setiap titik yang terdapat pada obyek, harus didefinisikan normal vektornya terlebih dahulu. Normal vektor setiap titik akan menyesuaikan arahnya berdasarkan matrik transformasi yang dikenakan pada titik

Implementasi Gouraud Shading

```
if (0.<normalzi) {  
    for (j=0;j<sphere.fc[i].NumberofVertices;j++) {  
        buff[j]=Vector2Point2D(vecbuff[j]);  
        NormalVector=unitVector(nrmvecbuff[j]);  
        colbuff[j]=PhongModel(LightVector,NormalVector,ViewVector,ColCyan);  
    }  
    gradatePolygon(buff,colbuff,sphere.fc[i].NumberofVertices);  
}
```

Jumlah warna pada setiap face sama dengan jumlah titik yang terdapat pada face tersebut.