

# 3D Photorealism

- ❑ Ketepatan pemodelan objek
- ❑ Proyeksi secara perspektif
- ❑ Efek pencahayaan yang natural kepada permukaan tampak: pantulan, transparansi, tekstur, dan bayangan



# Illumination model vs surface rendering

---

- Model pencahayaan: model untuk menghitung intensitas cahaya pada satu titik pada suatu permukaan
  - Rendering permukaan: prosedur yang menerapkan model pencahayaan untuk mendapatkan intensitas semua titik pada seluruh permukaan tampak
-

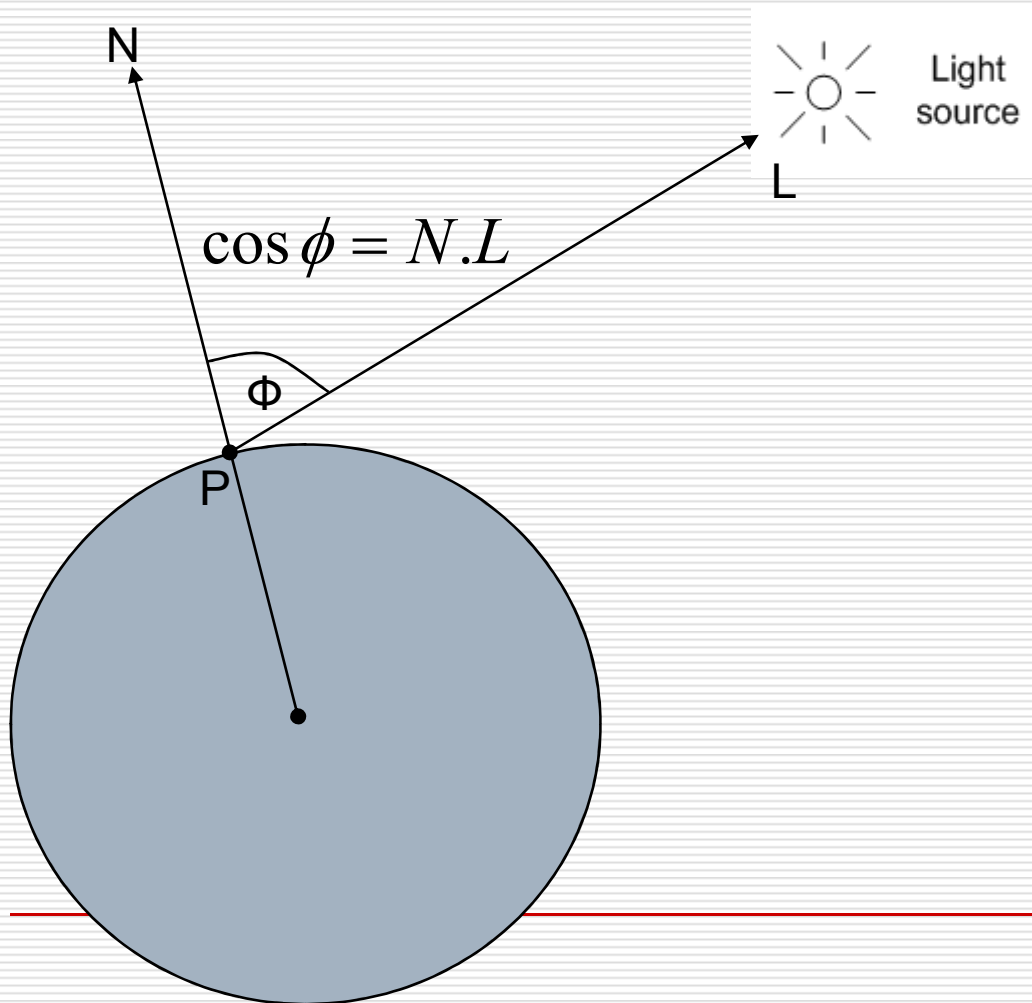
# Model Pencahayaan

---

- Ada 3 Jenis Warna yang mempengaruhi pewarnaan dalam proses rendering :
    - Ambient
    - Diffuse
    - Specular
-

# Model Pencahayaan

---



$$L = (L_x \ L_y \ L_z)$$

$$L_x = \frac{(X_L - X_P)}{|L|}$$

$$L_y = \frac{(Y_L - Y_P)}{|L|}$$

$$L_z = \frac{(Z_L - Z_P)}{|L|}$$

$$I_a = (W_a + W_d) \cdot K_a$$

$$I_d = W_d \cdot K_d \cdot \frac{\text{dist} \cdot \cos \phi}{\sqrt{\text{dist}}}$$

$$I_s = W_s \cdot \frac{\text{dist} \cdot (\cos \phi)^{K_s}}{\sqrt{\text{dist}}}$$

$$I_{total} = I_a + I_d + I_s$$

---

# Teknik Rendering Permukaan

---

- Melakukan perhitungan dengan model pencahayaan untuk semua titik yg tampak
    - Ray-tracing
  - Melakukan interpolasi untuk titik-titik pada permukaan dari sekumpulan intensitas hasil perhitungan dengan model pencahayaan
    - Scan-line
-

# ~~Scan-line algorithms~~

---

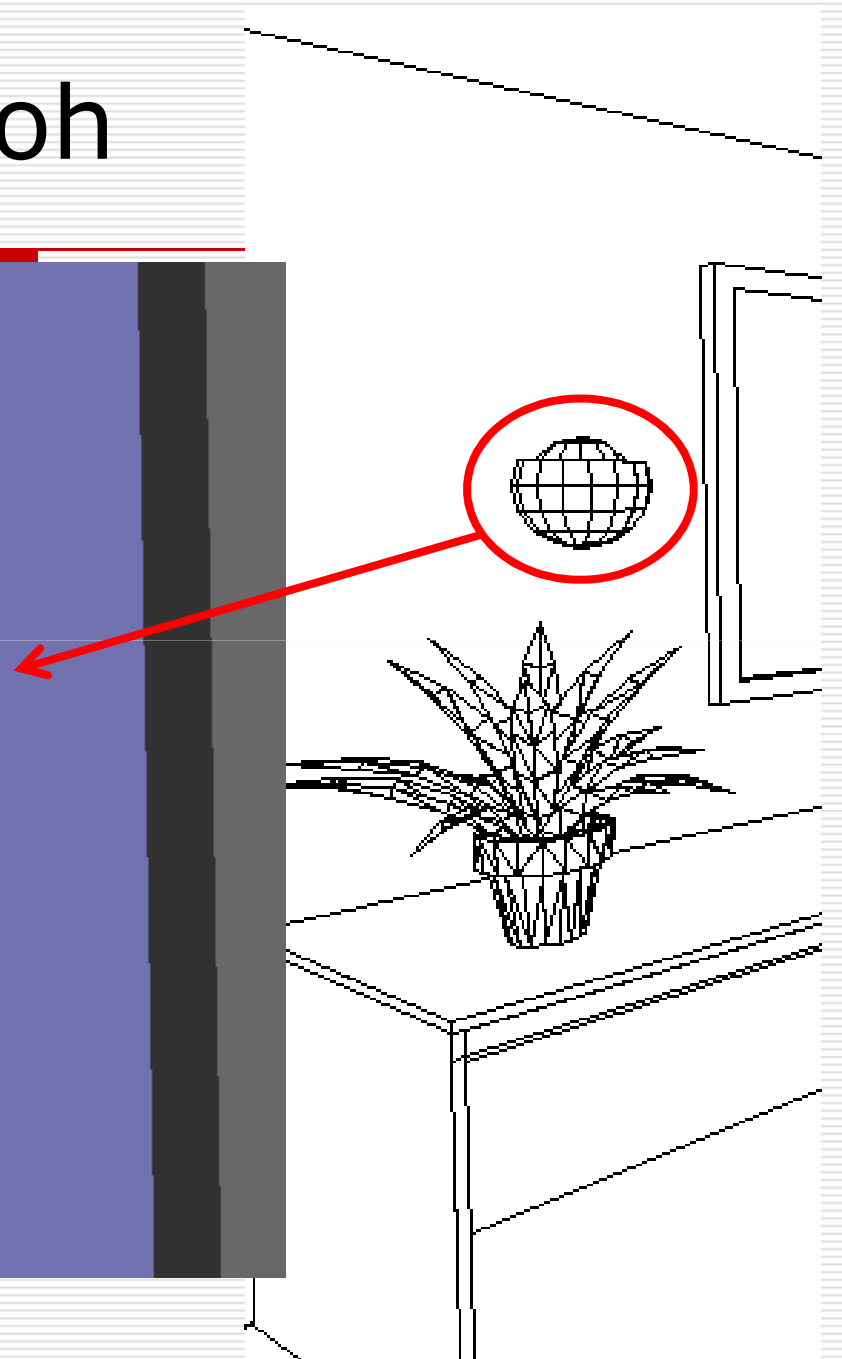
- Permukaan = poligon
  - Aplikasi model pencahayaan:
    - Perhitungan intensitas tunggal untuk masing-masing poligon
    - Intensitas tiap titik pada poligon didapat dengan cara interpolasi
  - Algoritma:
    - Flat (constant-intensity) shading
    - Gouraud shading
    - Phong shading
-

# Flat shading

---

- Intensitas tunggal untuk setiap poligon
  - Semua titik dalam poligon ditampilkan dengan intensitas yang sama
- Sering digunakan untuk mendapat tampilan cepat dari objek
- Akurat dengan asumsi:
  - Objek = polihedron (bukan aproksimasi kurva)
  - Sumber cahaya cukup jauh (N.L konstan)
  - Pengamat cukup jauh (V.R konstan)
- Bisa disiasati dengan memperkecil poligon facet

# Flat shading: contoh





# Gouraud shading

---

- Rendering poligon dengan interpolasi linear terhadap nilai-nilai intensitas vertex (titik sudut poligon)
  - Nilai intensitas untuk tiap poligon disesuaikan dengan poligon lain yang bersebelahan untuk mengurangi discontinuity (seperti yg terjadi pada flat shading)
-

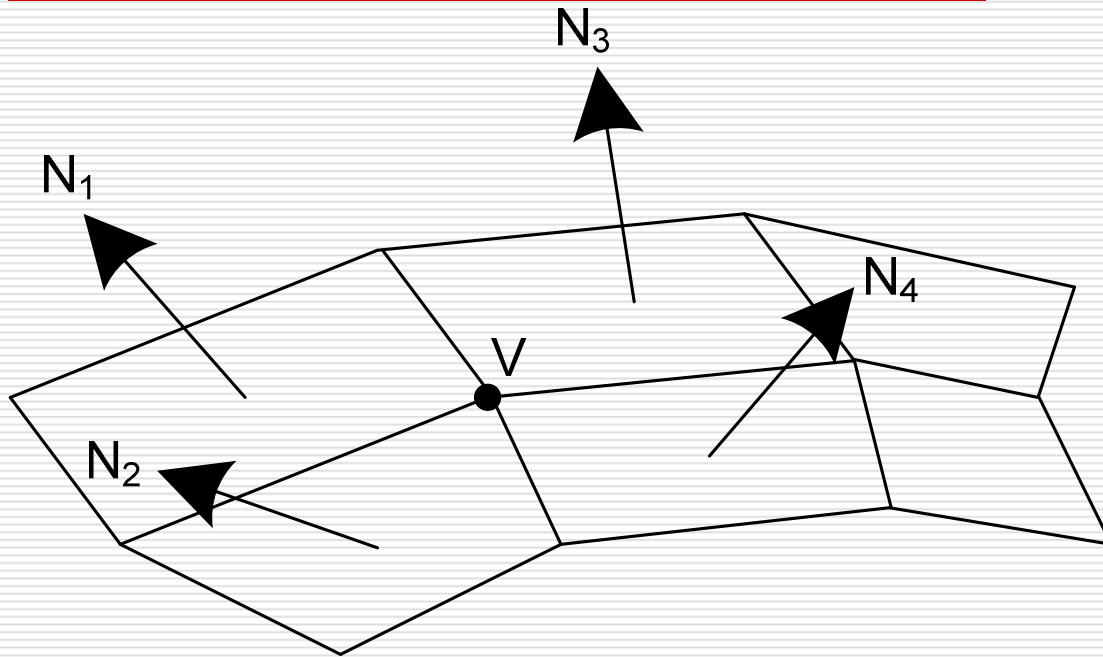
# Langkah-langkah Gouraud shading

---

- ❑ Tentukan vektor normal satuan rata-rata untuk setiap vertex pada poligon
  - ❑ Terapkan model pencahayaan ke tiap vertex untuk mendapatkan intensitasnya
  - ❑ Lakukan interpolasi linear terhadap intensitas vertex untuk mendapatkan intensitas titik-titik lain pada poligon.
-

# 1) Vektor normal untuk vertex V

---

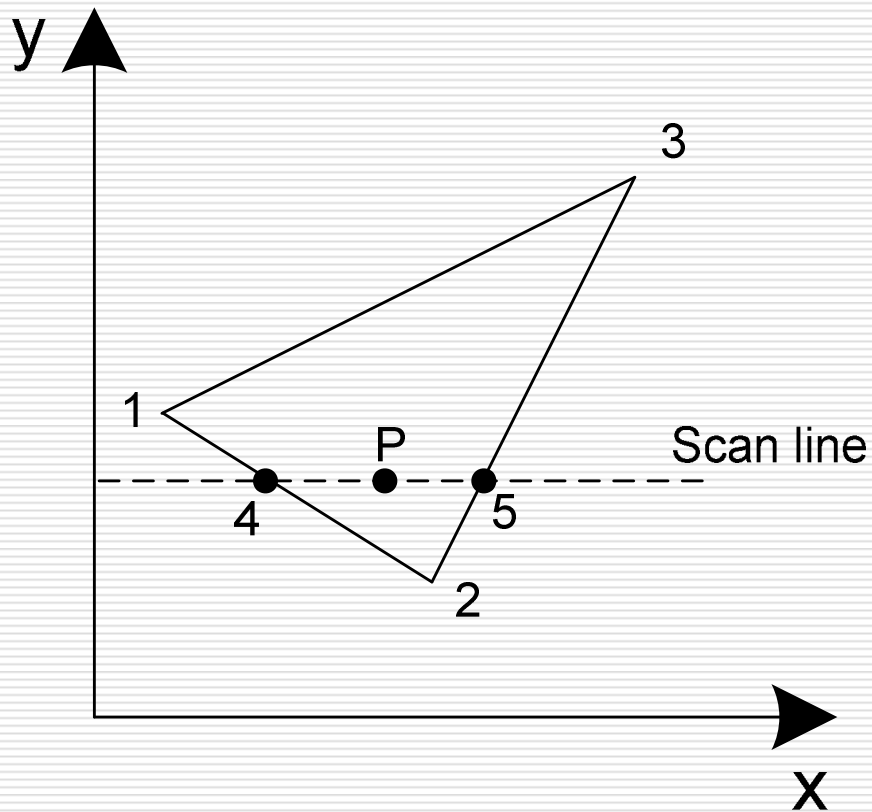


$$N_v = \frac{\sum_{k=1}^n N_k}{\left| \sum_{k=1}^n N_k \right|}$$

- Setelah mendapatkan vektor normal pada vektor V, dengan model pencahayaan bisa ~~didapat intensitas untuk titik tersebut~~

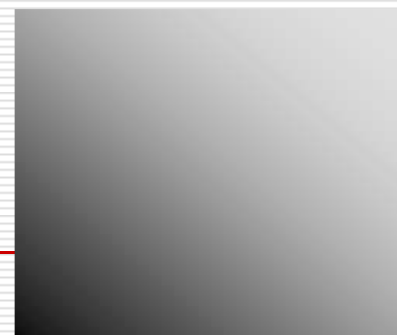
## 2) Interpolasi intensitas

---



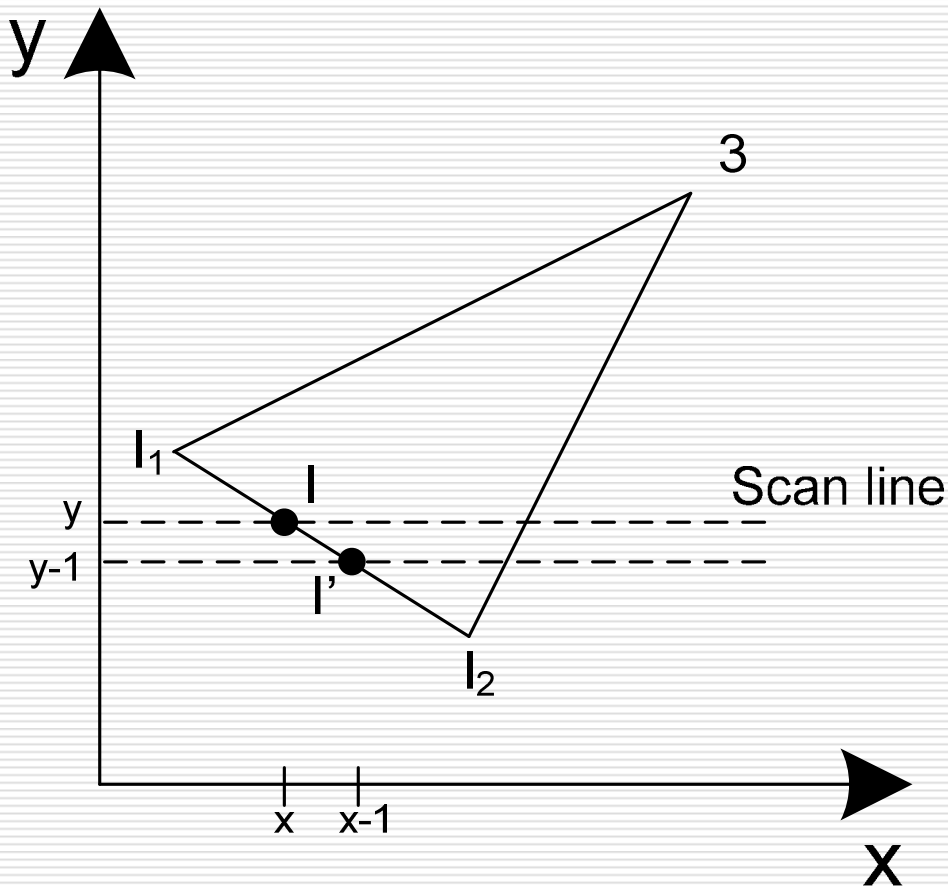
$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

$$I_P = \frac{x_5 - x_P}{x_5 - x_4} I_4 + \frac{x_P - x_4}{x_5 - x_4} I_5$$



# Interpolasi secara inkremental

---



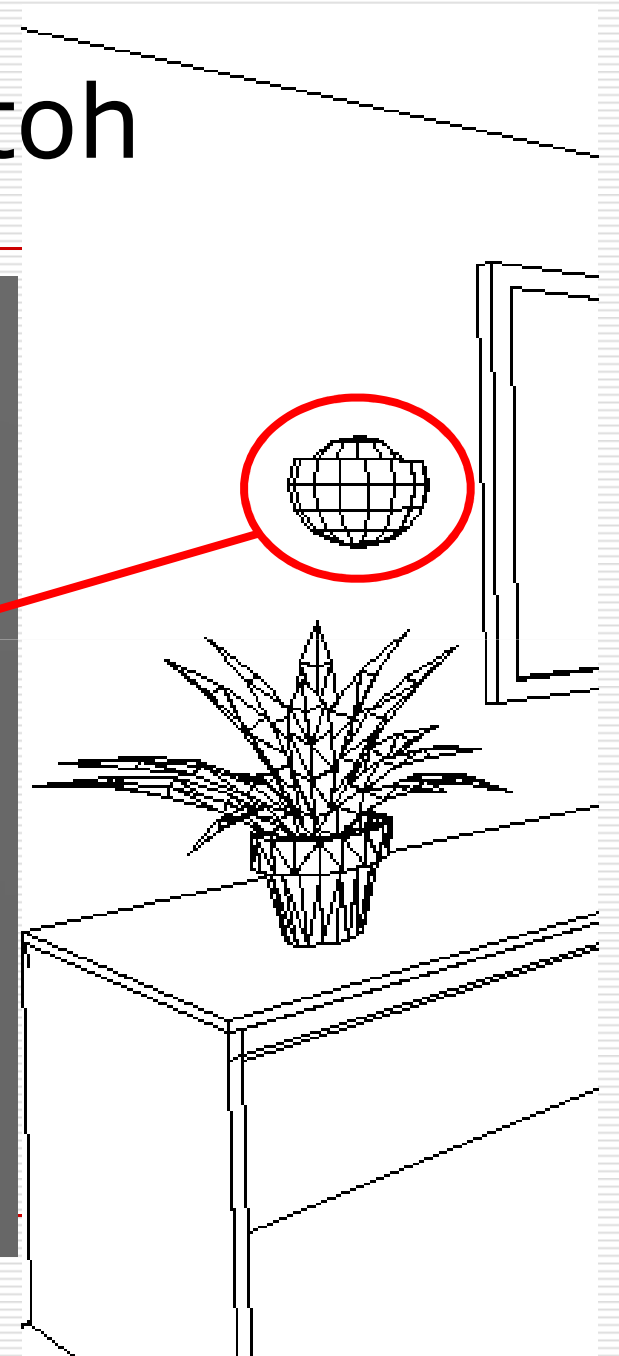
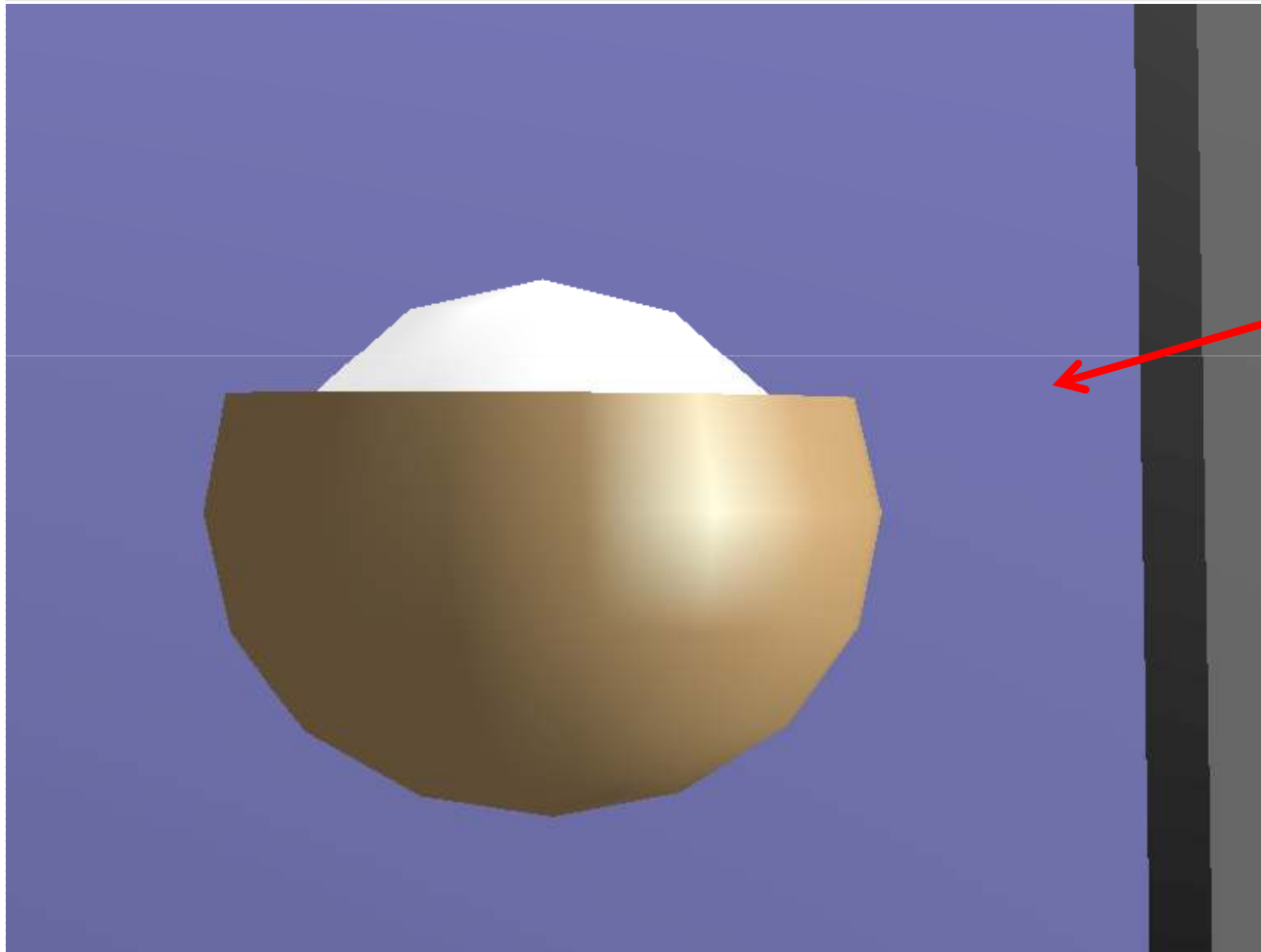
$$I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_2$$

$$I' = I + \frac{I_2 - I_1}{y_1 - y_2}$$

---

# Gouraud shading: contoh

---



# Kekurangan Gouraud

---

- ❑ Tampilan *highlight* tidak sempurna
  - ❑ Mach band: garis terang atau gelap muncul pada permukaan
    - Akibat penggunaan interpolasi linear
  - Untuk mengurangi efek tersebut:
    - Perkecil ukuran poligon
    - Gunakan metode lain (misal: Phong)
-

# Phong shading

---

- ❑ Interpolasi terhadap vektor normal
  - ❑ Model pencahayaan diterapkan pada semua titik pada permukaan
  - ❑ Memberikan *highlight* yang lebih realistik dan mereduksi efek Mach-band
-



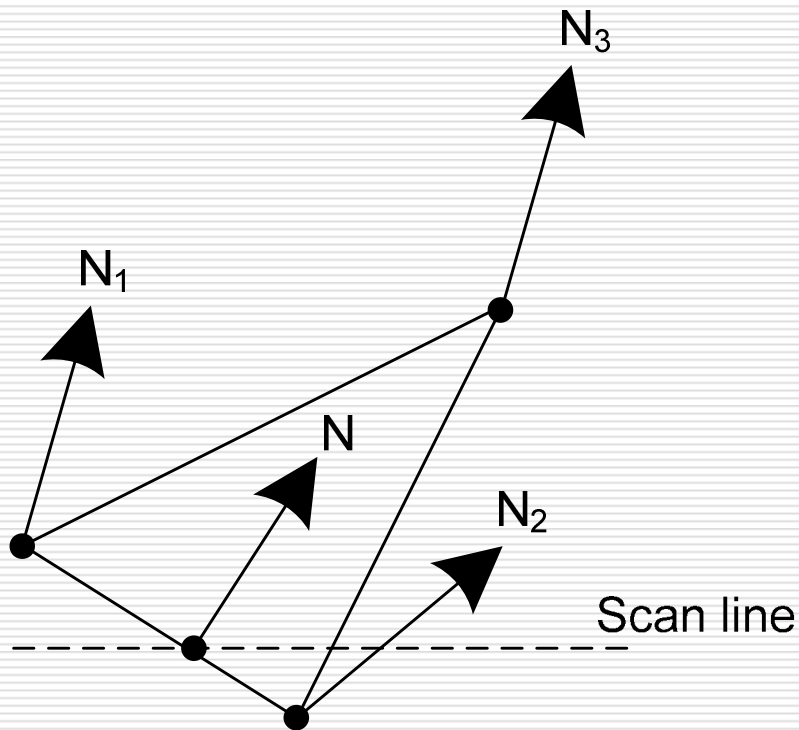
# Langkah-langkah Phong

---

- ❑ Tentukan vektor normal satuan rata-rata untuk setiap vertex pada poligon
  - ❑ Lakukan interpolasi linear terhadap vektor normal ke seluruh permukaan poligon
  - ❑ Terapkan model pencahayaan sepanjang scan line untuk mendapatkan intensitas setiap titik pada permukaan
-

# Interpolasi vektor normal

---

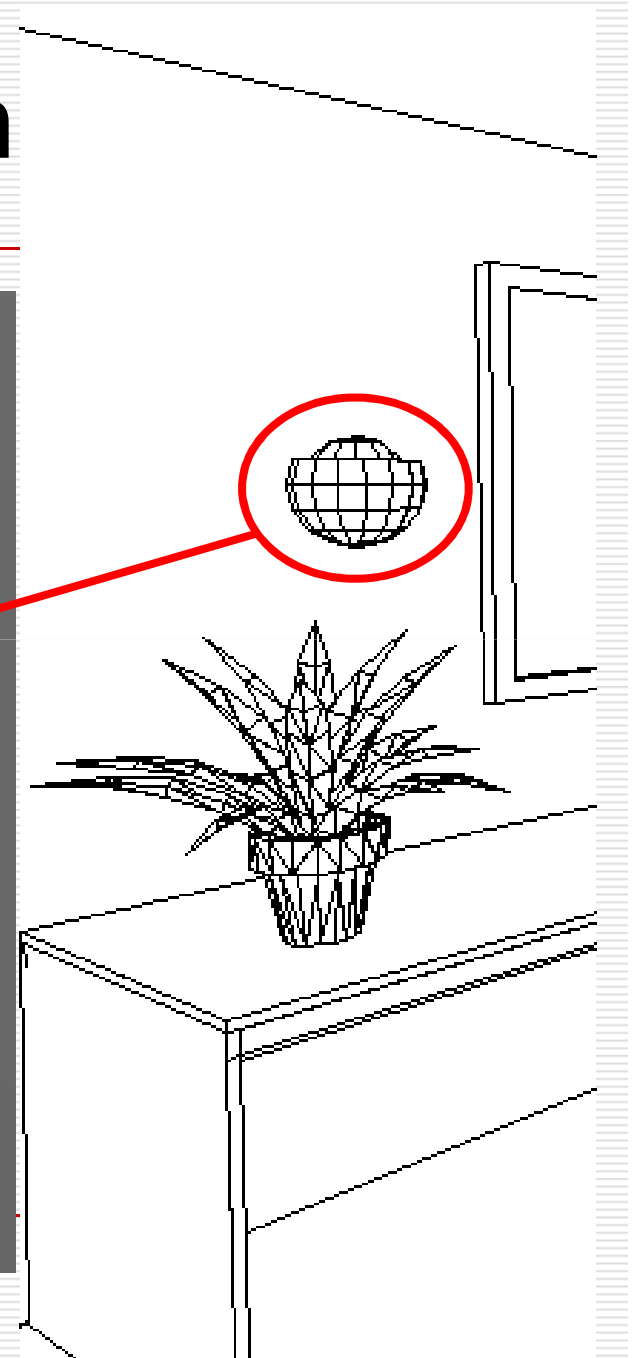
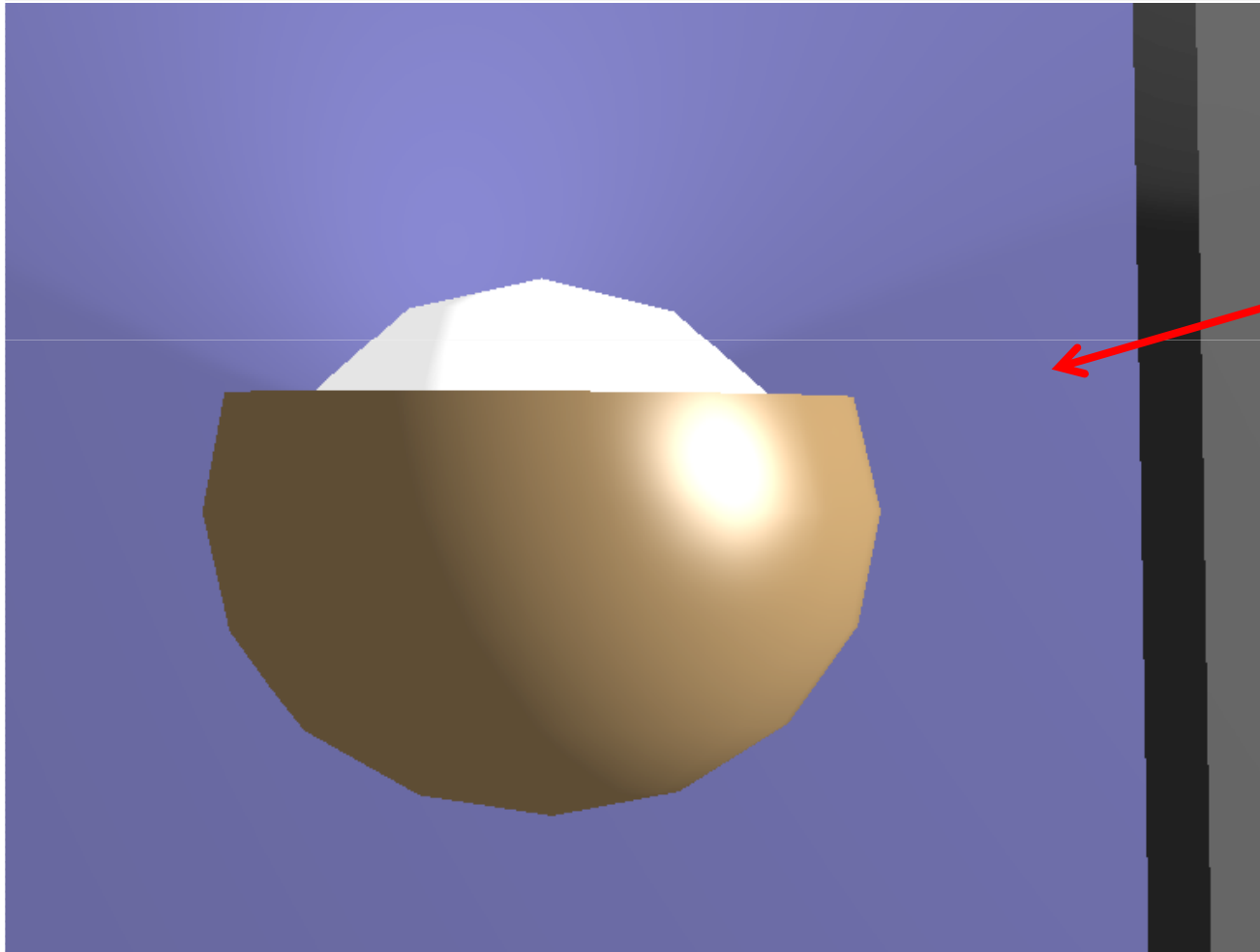


$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

- Untuk mendapatkan vektor-vektor normal antar scan line dan sepanjang scan line digunakan metode inkremental

# Phong shading: contoh

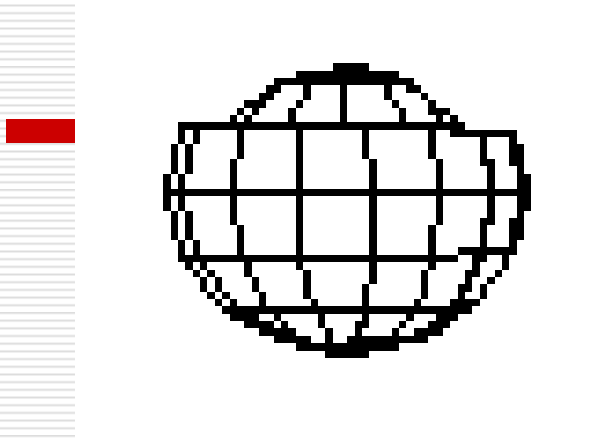
---



# Evaluasi Phong

---

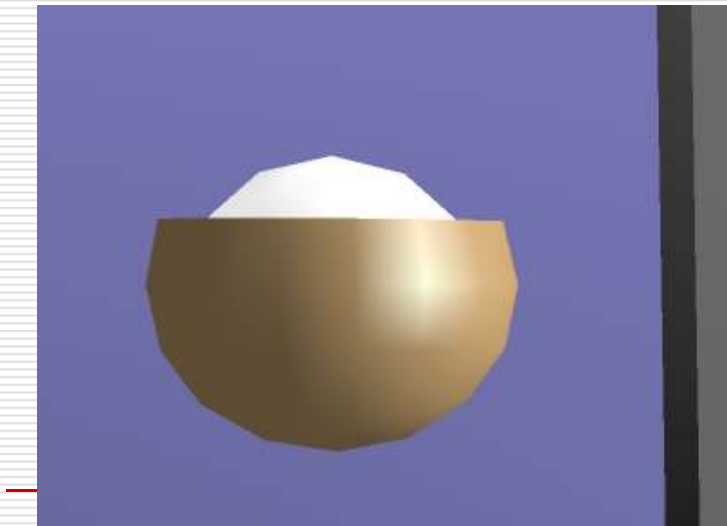
- Hasil lebih akurat
    - Interpolasi vektor normal
    - Model pencahayaan diterapkan pada tiap titik
  - Trade-off: butuh 'biaya' komputasi yang lebih besar
  
  - Fast Phong Shading:
    - aproksimasi intensitas dengan perluasan deret Taylor
    - permukaan dengan patch berbentuk segitiga
-



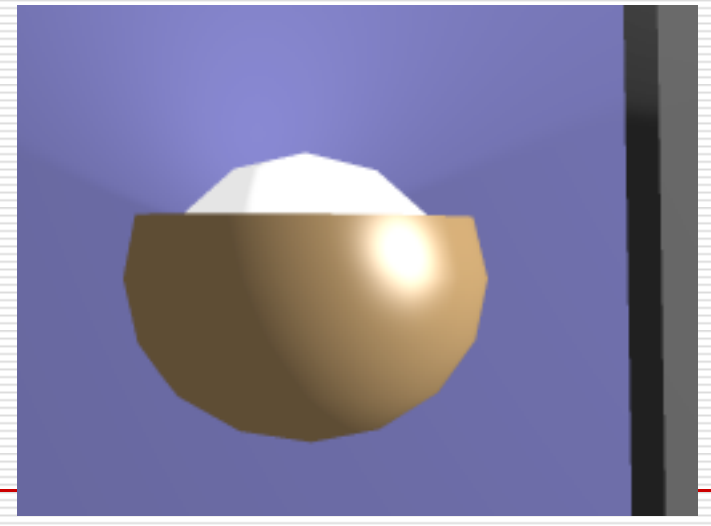
**wireframe**



**Flat shading**



**Gouraud shading**



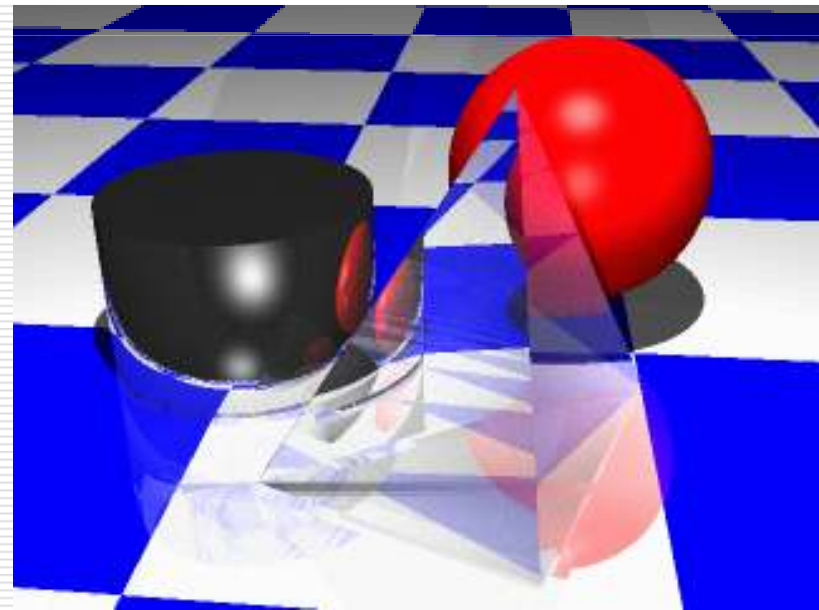
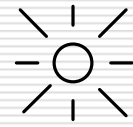
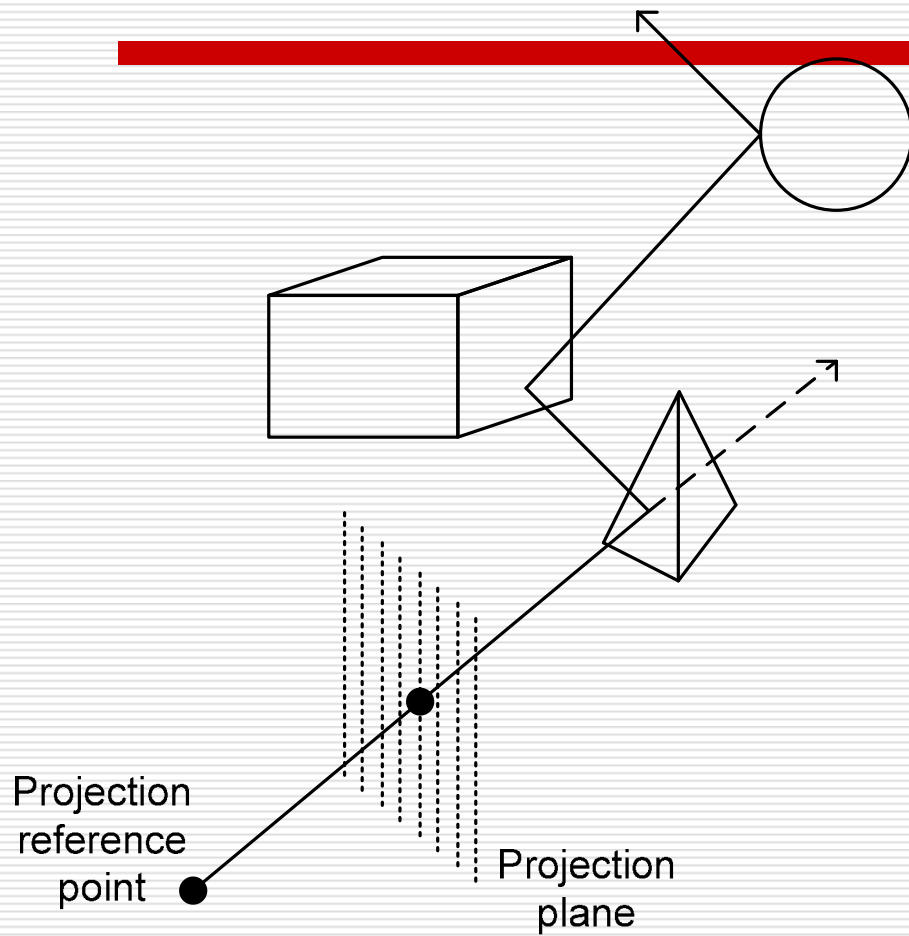
**Phong shading**

# Ray-tracing

---

- Kelanjutan ide Ray-Casting
    - 'Sinar' diteruskan (memantul ke / menembus objek lain)
    - Mencatat semua kontribusi terhadap intensitas suatu titik
    - Untuk mendapatkan efek pantulan dan transmisi secara global
  - Ray-Tracing dasar:
    - deteksi permukaan tampak, efek bayangan, transparansi, pencahayaan dengan beberapa sumber cahaya
  - Pengembangan Ray-Tracing:
    - tampilan fotorealistik (terutama objek mengkilap)
-

# Ilustrasi '*tracing a ray*'



# Algoritma Ray-Tracing Dasar

```
For each pixel in projection plane {  
    Create ray from the reference point passing through this pixel  
    Initialize NearestT to INFINITY and NearestObject to NULL  
  
    For every object in scene {  
        If ray intersects this object {  
            If t of intersection is less than NearestT {  
                Set NearestT to t of the intersection  
                Set NearestObject to this object  
            }  
        }  
    }  
  
    If NearestObject is NULL {  
        Fill this pixel with background color  
    } Else {  
        Shoot a ray to each light source to check if in shadow  
        If surface is reflective, generate reflection ray: recurse  
        If transparent, generate refraction ray: recurse  
        Use NearestObject and NearestT to compute shading function  
        Fill this pixel with color result of shading function  
    }  
}
```

---

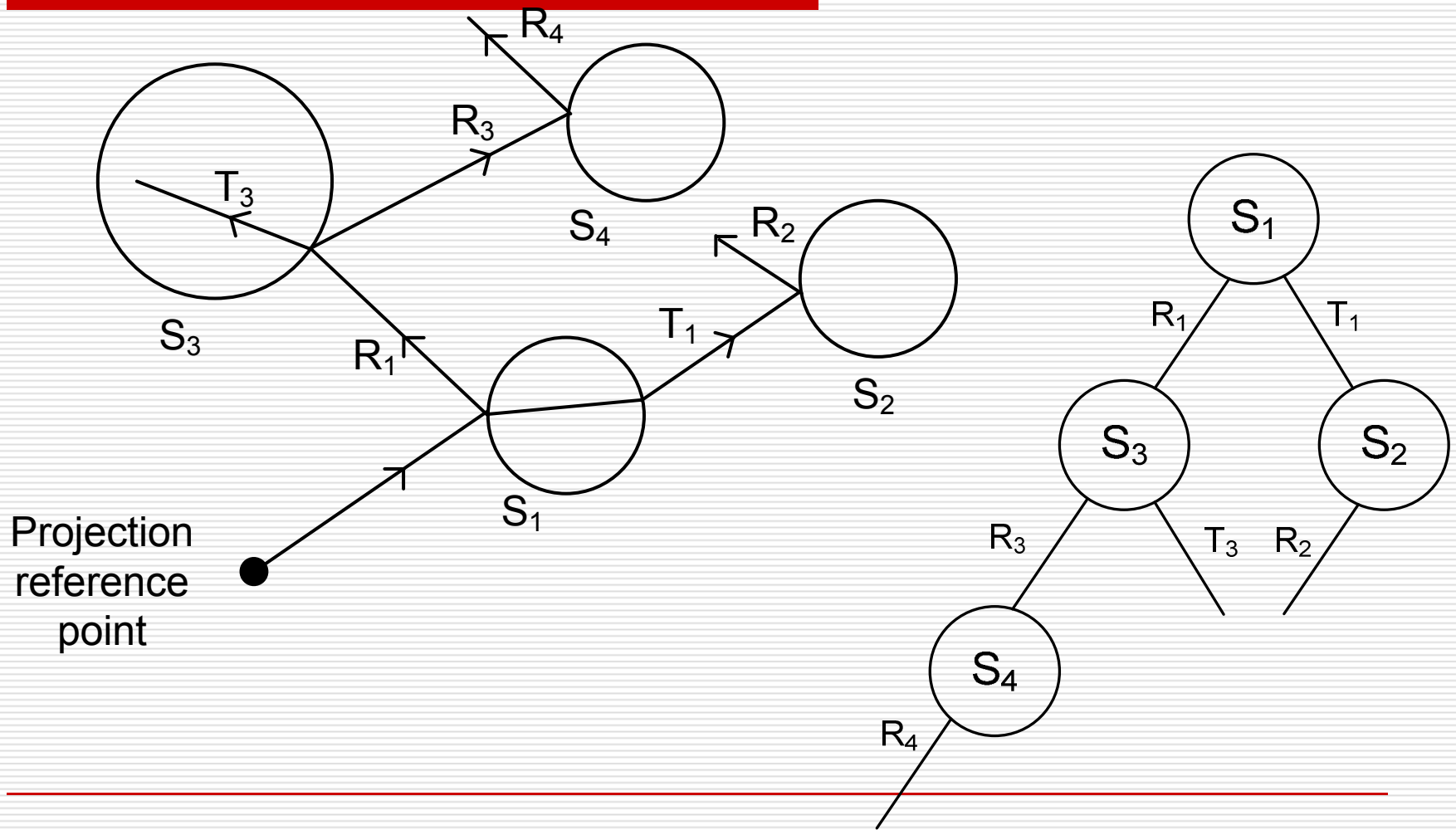


# Rekursif pada ray-tracing

---

- Saat *primary ray* (sinar yang berawal dari projection reference point) dipantulkan atau dibiaskan oleh objek, sinar pantulan atau biasan disebut dengan *secondary ray*
  - *Secondary ray* akan mengalami perlakuan yang sama seperti *primary ray* saat menemui objek (dipantulkan dan / atau dibiaskan)
-

# Binary Ray-Tracing tree



# Binary Ray-Tracing tree (cont'd)

---

- *Tracing* (pembentukan tree) berhenti jika:
  - Sampai *maximum depth* (pilihan user / kapasitas memori)
  - Sinar sampai ke sumber cahaya
- Intensitas pada suatu pixel: akumulasi intensitas mulai terminal node (paling bawah) pada tree
- Intensitas tiap permukaan mengalami atenuasi (pelemahan) setara dengan jarak permukaan tersebut ke permukaan *parent-nya* (pada tree: node yang tepat di atasnya)

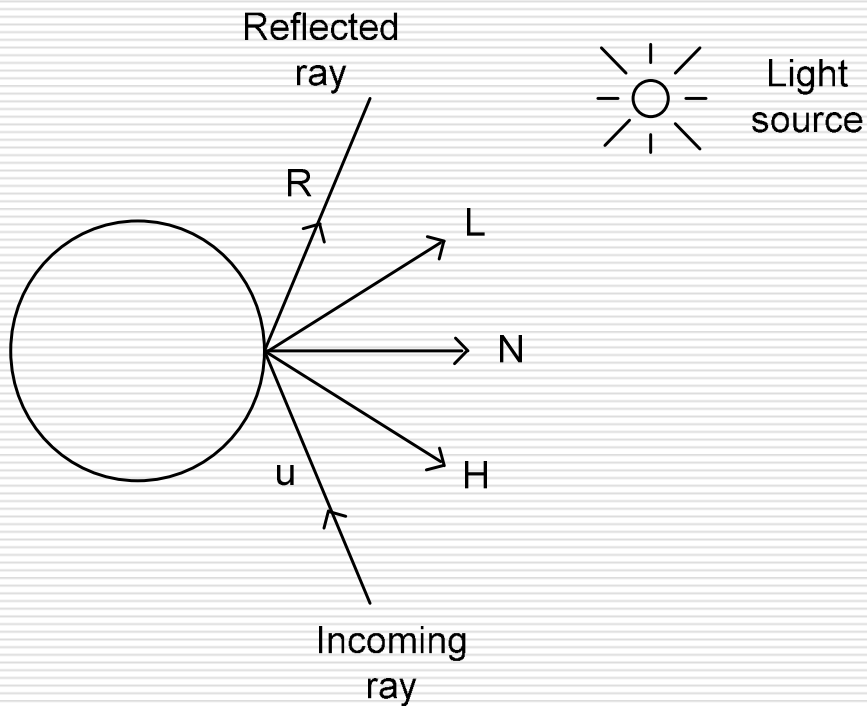
# Intensitas akhir suatu pixel

---

- Merupakan hasil penjumlahan seluruh intensitas -yang telah mengalami atenuasi- (pada root node)
  - Jika tidak ada permukaan yang berpotongan dengan sinar dari pixel, maka pixel tersebut diberi nilai intensitas sama dengan latar belakang
  - Jika sinar dari pixel berpotongan dengan sumber cahaya (meski tidak reflektif), maka pixel tersebut diberi nilai intensitas sama dengan sumber cahaya
-

# Pantulan

---



$$\textit{Ambient} = k_a I_a$$

$$\textit{Diffuse} = k_d (N \cdot L)$$

$$\textit{Specular} = k_s (H \cdot N)^{n_s}$$

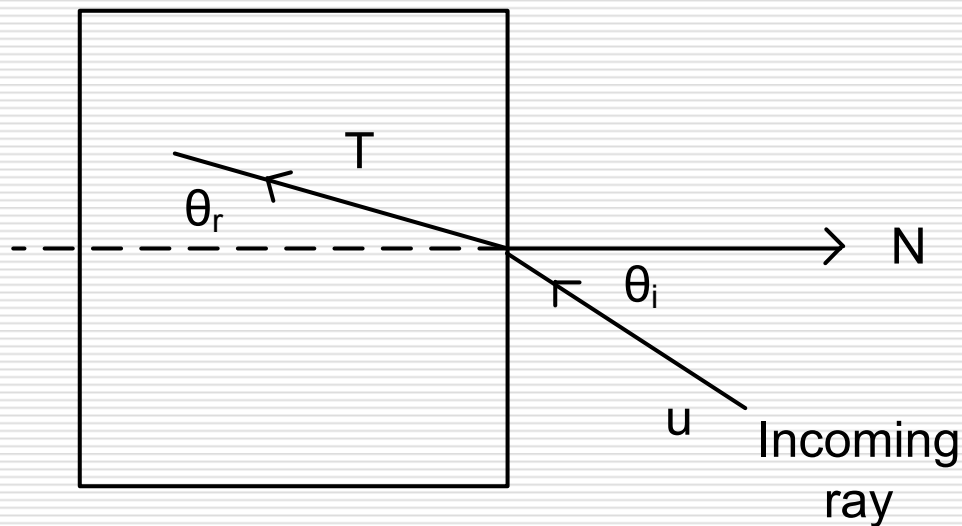
$$R = u - (2u \cdot N)N$$

- Jika  $L$  berpotongan dengan permukaan lain, maka permukaan tersebut dalam daerah bayangan

# Pembiasan

---

- Untuk objek dengan material transparan



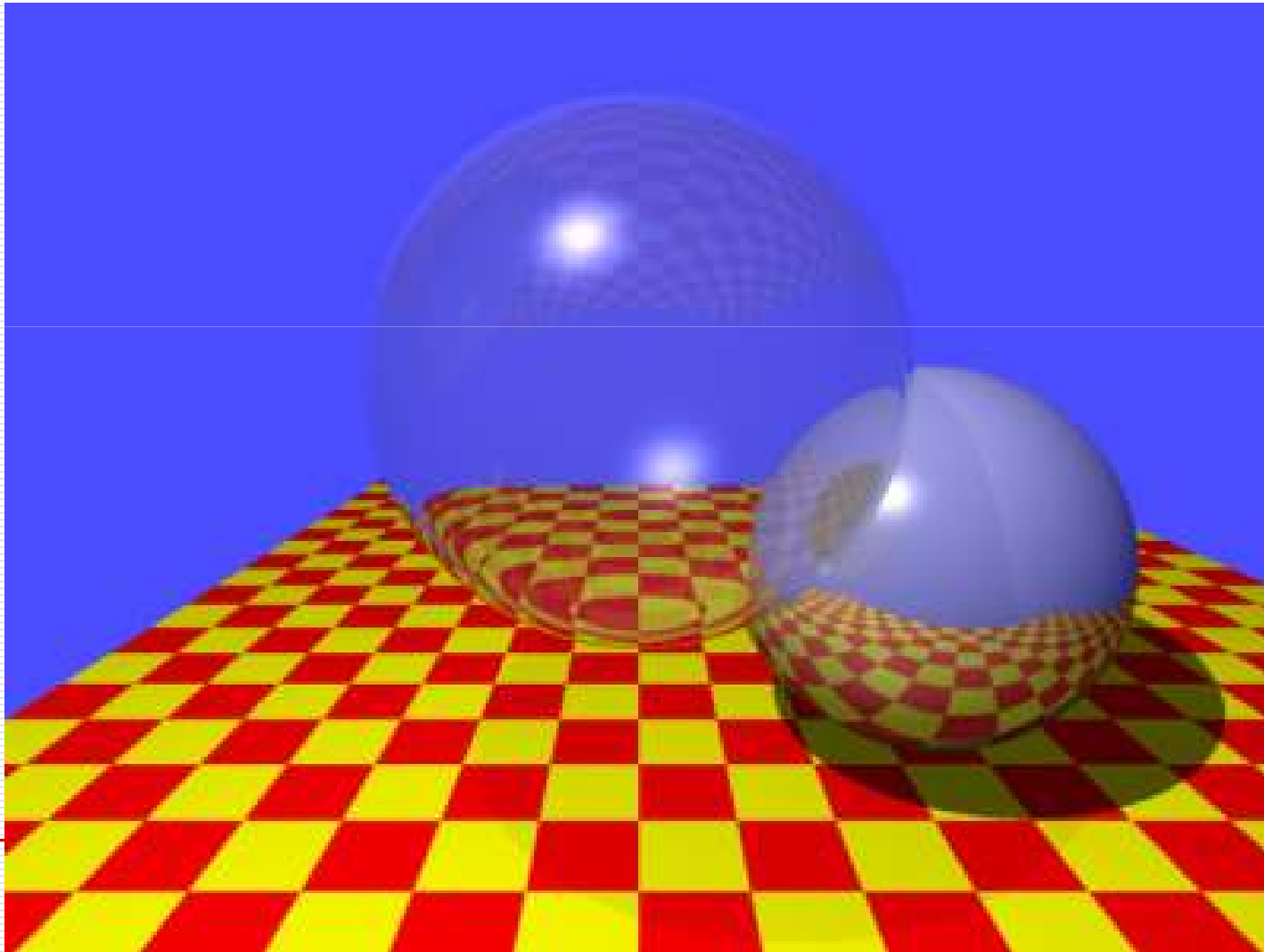
$$T = \frac{\eta_i}{\eta_r} u - \left( \cos \theta_r - \frac{\eta_i}{\eta_r} \cos \theta_i \right) N$$

$$\cos \theta_r = \sqrt{1 - \left( \frac{\eta_i}{\eta_r} \right)^2 (1 - \cos^2 \theta_i)}$$

---

# Contoh Ray-Tracing

---



# Radiosity

---

- ❑ Memodelkan pantulan difusi dengan lebih akurat
  - ❑ Mempertimbangkan transfer energi radian antar permukaan (sesuai dengan hukum kekekalan energi)
  - ❑ Tingkat kecerahan (*brightness*) dan warna dari segala sesuatu tergantung dari segala sesuatu yang lain
  - ❑ Lebih realistik
-



# Efek visual radiosity

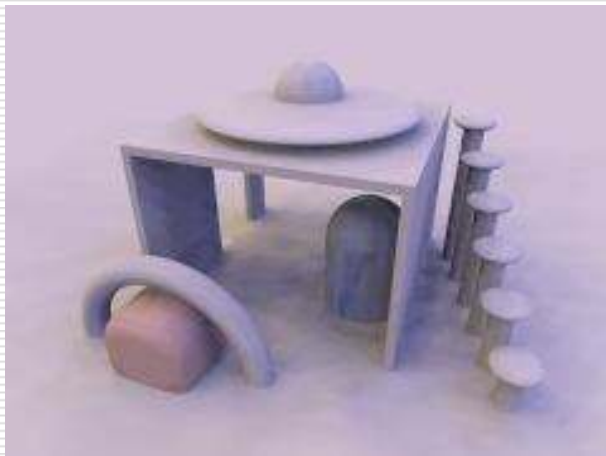
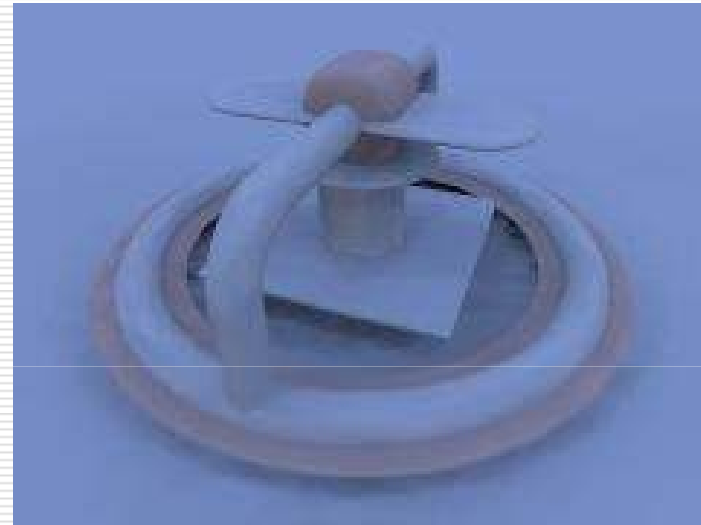
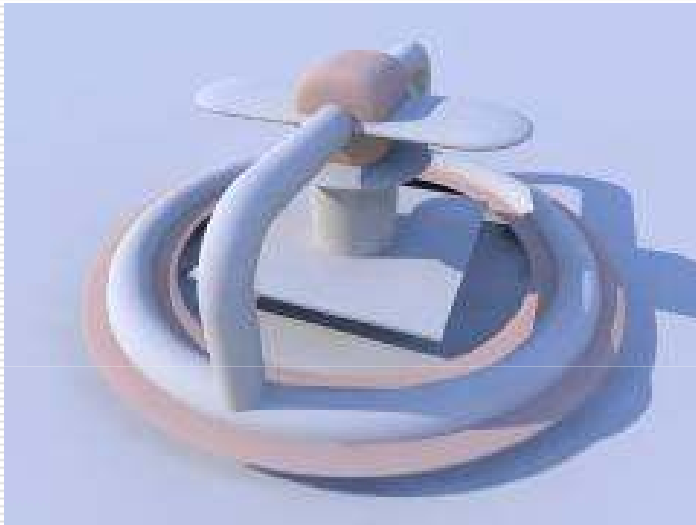
---



- Cahaya putih mengenai bola merah
- Ada pantulan cahaya merah dari bola ke objek lain di sekelilingnya
- ~~Lantai putih di sekitar bola menjadi kemerah-merahan~~

# Contoh lain radiosity

---



# Teori dasar radiosity

---

- Radiosity (B): energi per satuan luas yang meninggalkan permukaan per satuan waktu; total energi yang dipancarkan dan yang dipantulkan

$$B_i dA_i = E_i dA_i + R_i \int_j B_j F_{ji} dA_i$$

Radiosity x luas = energi dipancarkan + energi dipantulkan

---

# Teori dasar radiosity (cont'd)

---

- Hubungan timbal balik:

$$F_{ij}A_i = F_{ji}A_j$$

- Setelah dibagi dengan  $dA_i$ :

$$B_i = E_i + R_i \int_j B_j F_{ij}$$

- Untuk lingkungan diskrit:

$$B_i = E_i + R_i \sum_{j=1}^n B_j F_{ij}$$

---

# Teori dasar radiosity (cont'd)

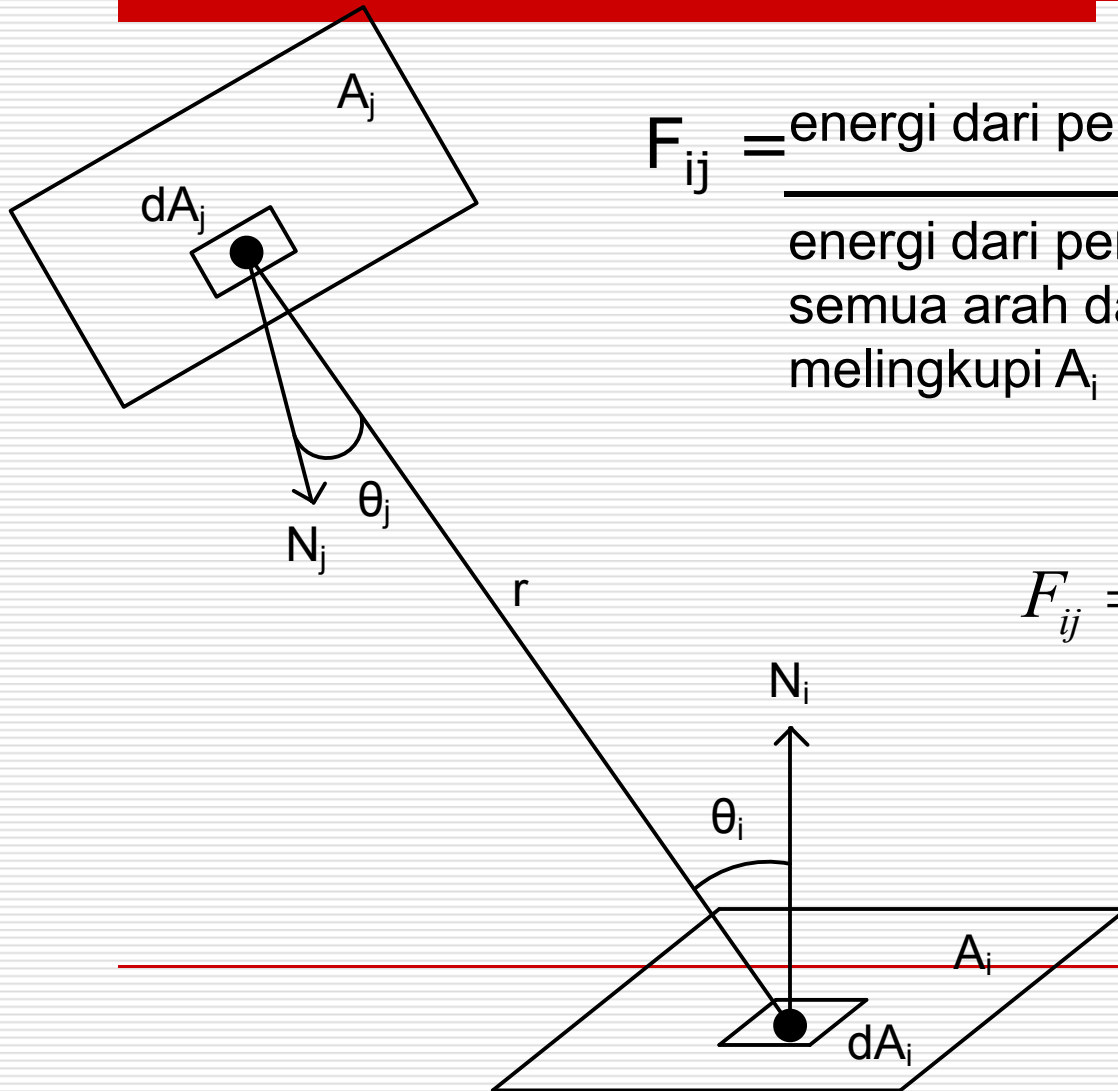
---

- Tiap permukaan saling mempengaruhi, sehingga perlu menyelesaikan n persamaan secara simultan:

$$\begin{bmatrix} 1-R_1F_{11} & -R_1F_{12} & \dots & -R_1F_{1n} \\ -R_2F_{21} & 1-R_2F_{22} & \dots & -R_2F_{2n} \\ \dots & \dots & \dots & \dots \\ -R_nF_{n1} & -R_nF_{n2} & \dots & 1-R_nF_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \dots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \dots \\ E_n \end{bmatrix}$$

- Radiosity bersifat monokromatik.
- Untuk RGB, lakukan perhitungan untuk tiap warna

# Form factor



$F_{ij}$  = energi dari permukaan  $A_i$  yang sampai ke  $A_j$   
energi dari permukaan  $A_i$  yang menyebar ke semua arah dalam ruang hemisphere yang melingkupi  $A_i$

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos \theta_i \cos \theta_j}{\pi r^2} dA_j dA_i$$

# Asumsi dlm perhitungan form factor

---

- Berlaku hukum kekekalan energi

$$\sum_{j=1}^n F_{ij} = 1$$

- Pantulan cahaya seragam

$$A_i F_{ij} = A_j F_{ji}$$

- Permukaan datar atau convex

$$F_{jj} = 0$$

---