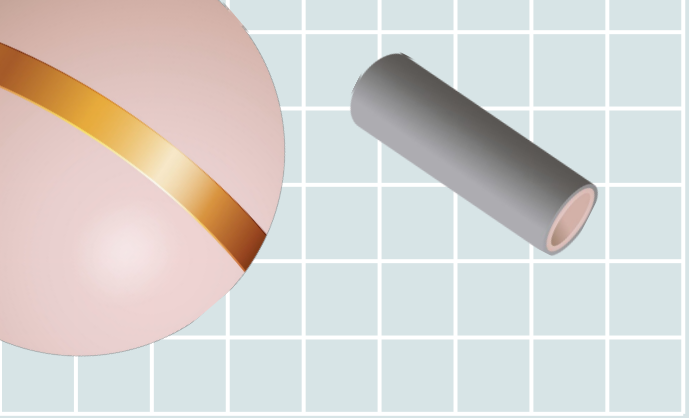
Elaborato di Calcolo Scientifico per l'Innovazione Tecnologica

Lighting & Shading il modello di Phong

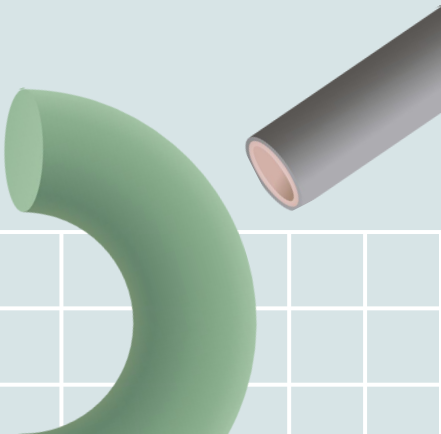
Un lavoro di:
Stefano Marano, Russo Paolo, Vito Romano e Marco Cimmino

Indice



01

Cenni sulla Computer Graphics

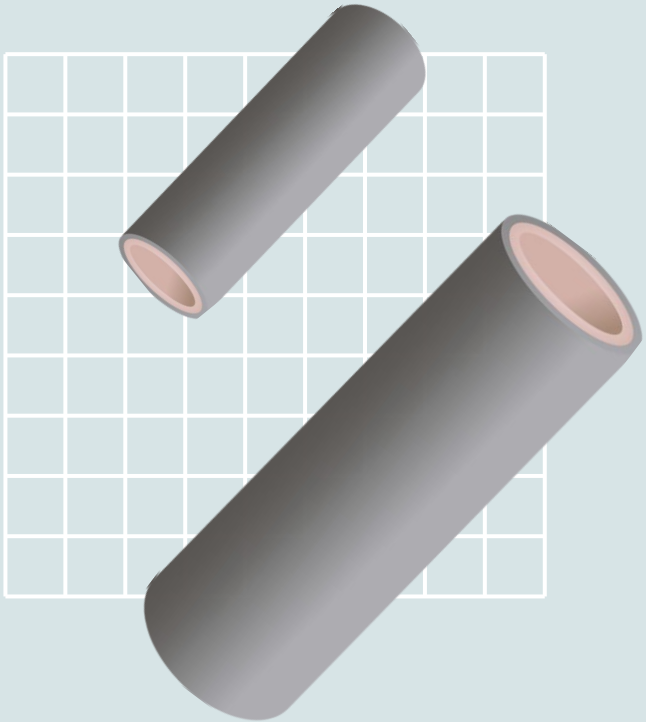


Cos’è la Computer Graphics?

La Computer Graphics è un campo dell'informatica e della matematica che si occupa della **creazione**, **manipolazione** e **rappresentazione** visuale di immagini, video e oggetti in 2D e 3D utilizzando un computer.

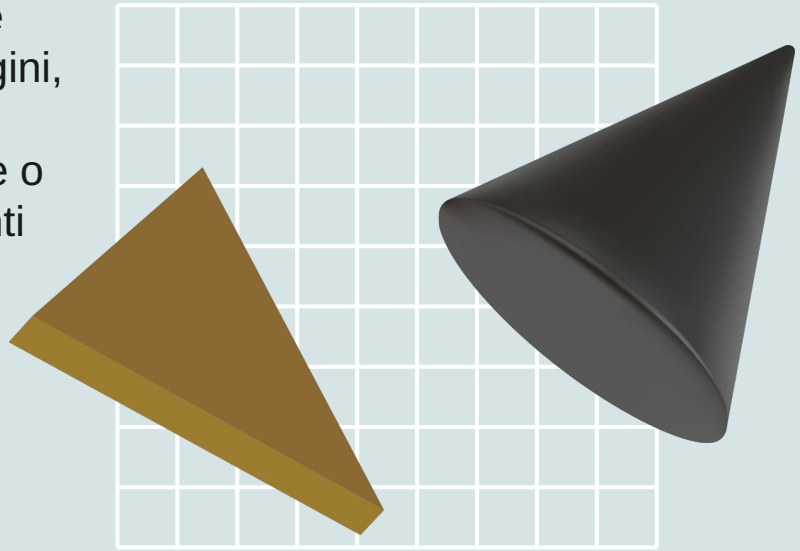
Questo campo si occupa di sviluppare algoritmi, tecniche e software per generare grafica computazionale, che può essere utilizzata in varie applicazioni come videogiochi, animazioni, simulazioni, rendering di immagini realistiche, grafica per il web e molto altro.

La computer graphics combina principi di matematica, fisica, percezione visiva e programmazione per creare e gestire le immagini generate al computer.



Cos’è la image analysis?

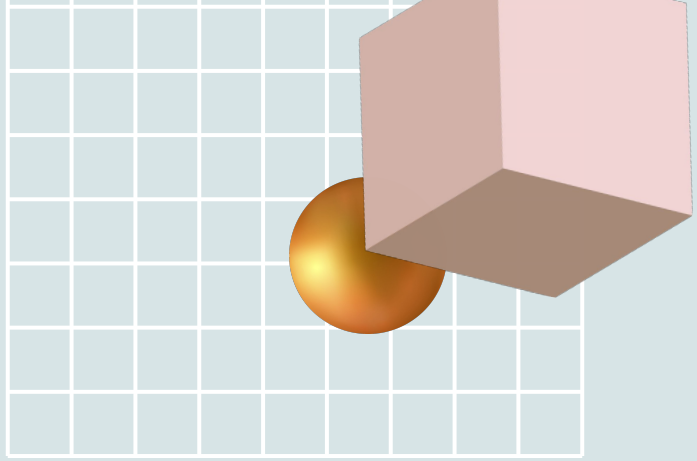
La image analysis si occupa di elaborare e interpretare le informazioni contenute in un'immagine, attraverso algoritmi e tecniche di elaborazione delle immagini, vengono estratte **caratteristiche** visive e strutturali dalle immagini, che possono essere utilizzate per scopi di riconoscimento, classificazione, localizzazione o misurazione di oggetti o caratteristiche presenti nell'immagine stessa.



Questo processo di analisi delle immagini può essere utilizzato in diversi settori, come la medicina, l'automazione industriale, la videosorveglianza e molti altri.

Non va confusa con la computer graphics !

Aspetti generali



Metodologie

Sono di natura matematica e fisica, si utilizza l’algebra, la geometria e l’ottica;
Ad esempio: viene sfruttata la legge di **Lambert** per l’illuminazione locale.

Algoritmi

- Algoritmi di **Rendering**, per generare scene in 3D.
- Algoritmi di **simulazione**.
- Algoritmi di **illuminazione**: sono utilizzati per calcolare l'illuminazione e le ombre in una scena 3D

Passi fondamentali

Modeling

Acquisire i dati e posizionare l’oggetto nella scena.

Shading

Posizionare le luci e definire l’illuminazione

Viewing

Definire il punto di vista

Clipping

Specificare le parti visibili

Proiezione

Proiettare la scena 3D sul piano immagine 2D

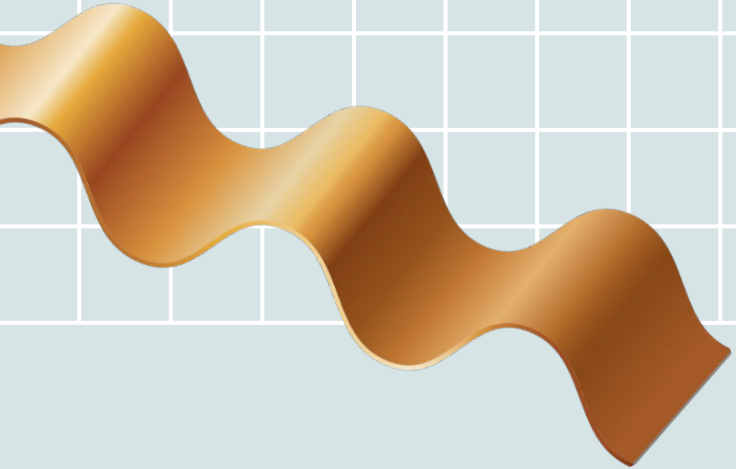
Rasterization

Conversione in immagine attraverso matrici di pixel

Lighting e Shading

Il Lighting e lo Shading sono strumenti di fondamentale importanza per far apparire le immagini grafiche più **realistiche** e **comprensibili**; essi forniscono, inoltre, indicazioni visive circa la curvatura e l'orientamento delle superfici oltre, naturalmente, ad evidenziare la tridimensionalità di un'immagine grafica;

Nell'ambito della Computer Graphics, i metodi di illuminazione ed ombreggiatura si basano su approcci modulari in cui, esplicitamente, il programmatore specifica le posizioni e le proprietà delle sorgenti luminose e dei materiali;



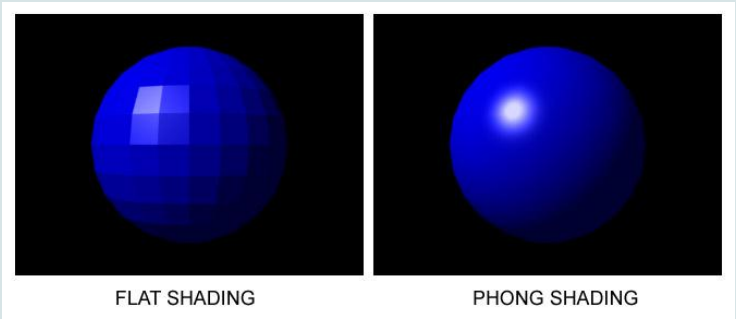
Modelli di Riferimento

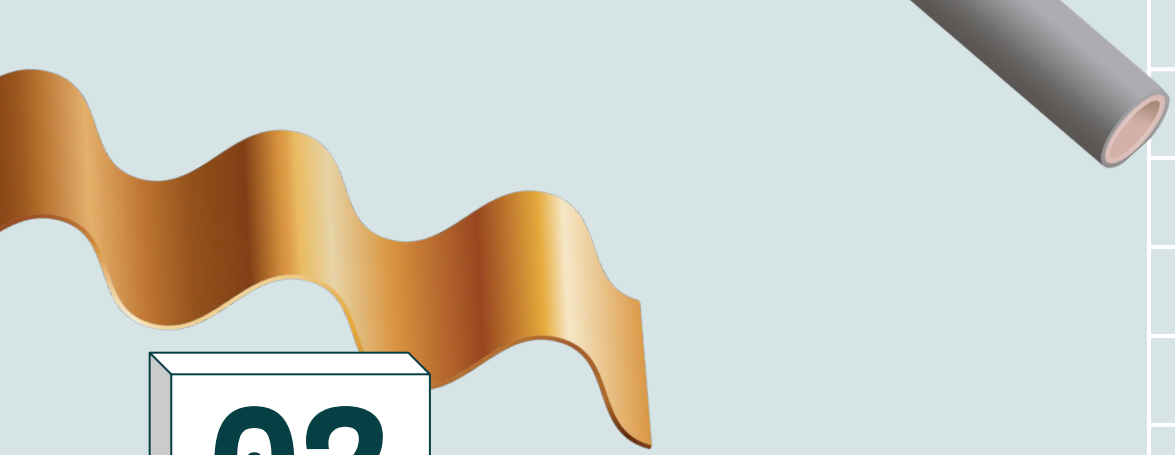
I modelli d’illuminazione più diffusi sono:

- Modello di Phong (1975);
- Modello di Cook-Torrance (1982);

I modelli di ombreggiatura più diffusi sono:

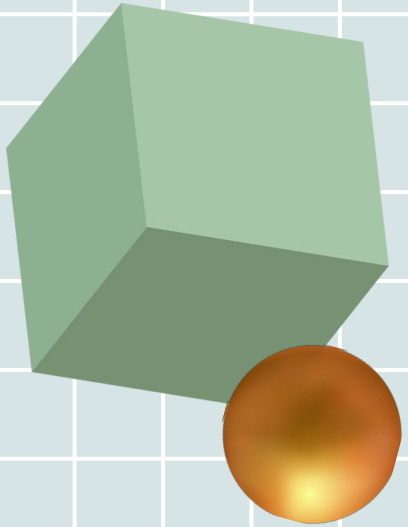
- Shading di Gouraud (1971);
- Shading di Phong (1975);





02

Modello di Phong

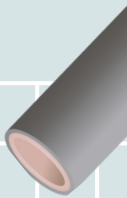


Modello di illuminazione locale

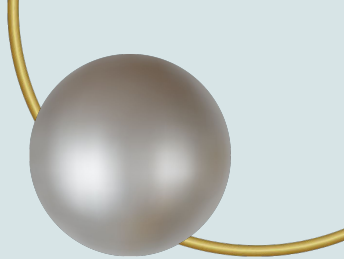
Definizione Il modello di illuminazione locale descrive l’interazione luce-materia localmente in un punto della superficie. Esso considera:

- Sorgenti di luce puntiforme
- Illuminazione ambientale costante
- Riflessione diffusiva
- Riflessione speculare

Il **modello di Phong** è un modello empirico di illuminazione locale dei punti su una superficie.



Modello di illuminazione di Phong



Il modello di riflessione di Phong fu sviluppato da Bui Tuong Phong all'Università dello Utah, il quale lo pubblicò nella sua dissertazione di dottorato nel 1975.
I metodi di Phong sono stati considerati radicali al tempo della loro introduzione e da allora sono diventati la *de facto* linea base per i metodi di shading di molte applicazioni di rendering.
I metodi di Phong sono diventate popolari grazie al loro uso generalmente **efficiente** del tempo di computazione per pixel renderizzato, ma mantenendo una grande **flessibilità** applicativa e **semplicità** di applicazione, sia in software che in hardware.
Fisicamente, sfrutta le proprietà di riflessione della luce sulle superfici a partire da sorgenti luminose che vengono modellate come sorgenti puntiformi;

L’unico fenomeno fisico modellato è la **riflessione diretta**, non ci occupiamo della modellazione della rifrazione, quindi le equazioni che vedremo non riescono a simulare il comportamento di materiali trasparenti o semitrasparenti.

Modello di illuminazione di Phong - Aspetto fisico

Esso descrive il modo in cui la superficie riflette la luce, come una combinazione degli effetti sulle superfici **opache** con le caratteristiche delle superfici **lucide**.
Infatti prevede due tipologie di riflessione:

- **Riflessione diffusa**: la luce riflessa in modo diffuso è quella che viene riflessa uniformemente in tutte le direzioni dalla superficie, come accade nelle superfici **opache**.
- **Riflessione speculare**: la luce riflessa specularmente è quella che viene riflessa in modo specchiato, come accade nelle superfici **lucide**, quindi non riflette con uguale intensità in tutte le direzioni, ma dipende dall'osservatore.

$$E_{tot} = E_D + E_a + E_s$$

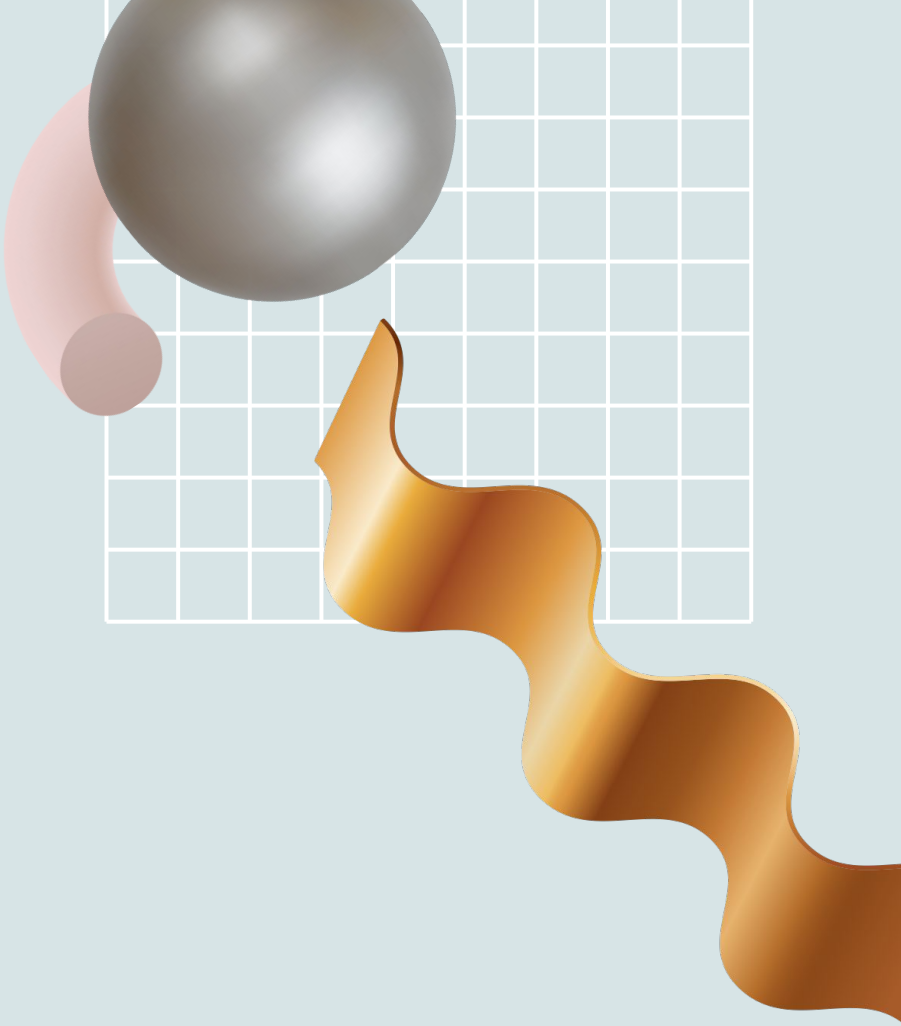
È basato sull'informale osservazione di Phong per cui le superfici lucide presentano piccoli intensi punti luce, mentre le superfici smussate presentano punti luce più grandi che decadono più gradualmente, inoltre si introduce un termine di **riflessione ambientale** per tenere conto della piccola quantità di luce che si disperde nell'intera scena.

Luce ambientale

Oggetti illuminati con un modello di illuminazione comprendente solo il termine di illuminazione diffusa e speculare risultano non troppo realistici, come se fossero illuminati da una torcia in un ambiente altrimenti completamente oscuro

La reciproca riflessione diffusa tra gli oggetti della scena viene approssimata da una componente ambiente, che si assume distribuita uniformemente, con intensità L_a uguale in ogni punto della superficie.

$$E_a = K_a \cdot L_a$$

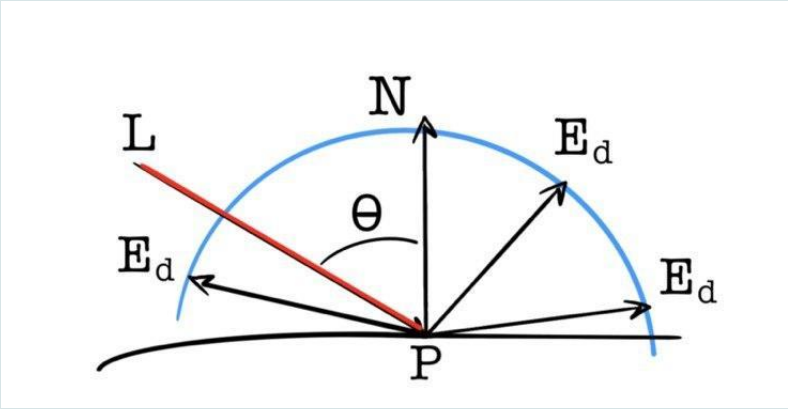


Riflessione diffusa

La riflessione diffusa prevede che la luce venga riflessa in modo uguale in tutte le direzioni, quindi la luminosità dipende solo dall'angolo θ formato dalla direzione del raggio luminoso (L) e la normale alla superficie nel punto di incidenza (N).
In queste condizioni la quantità di luce che arriva all'osservatore è data dalla legge di Lambert.

Legge di Lambert - $E_D = K_D L \cos\theta$

- K_D è il coefficiente di riflessione diffusa, approssima il grado di diffusività della superficie, cioè la frazione di luce incidente che viene riflessa
- L è la direzione di incidenza della sorgente di luce, quindi rappresenta l'intensità del raggio incidente in P.
- N è la normale alla superficie



Riflessione diffusa

Legge di Lambert - $E_D = K_D\, L \cos\theta$

L’angolo θ deve avere un valore compreso tra 0° e 90° per contribuire all’illuminazione del punto, in altre parole un punto non è illuminato da luci che stanno dietro di esso, quindi si riesce a modellare la diversa distribuzione della luce in base all’incidenza della luce.

Inoltre se i vettori N e L sono normalizzati si può riscrivere l’equazione utilizzando il loro prodotto scalare e per tener conto anche dell’attenuazione dell’intensità dell’illuminazione con la distanza si introduce anche un fattore di attenuazione F_{att}.

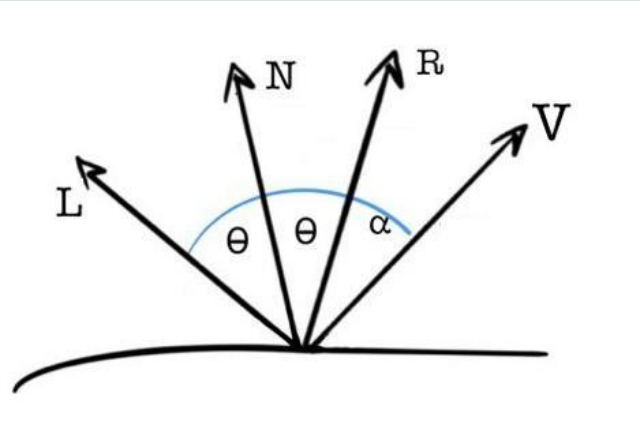
$$E_D = F_{att}\, K_D\, L\, (\overline{L} \cdot \overline{N})$$

Riflessione speculare

Da una superficie totalmente lucida, la luce viene riflessa solo nella direzione di riflessione R che, geometricamente, non è altro che L (direzione di incidenza) riflessa rispetto alla normale della superficie. L'osservatore può vederla solo se la direzione di vista è allineata con la riflessione, cioè se l'angolo α tende a 0.

$$E_S = L\, K_S \cos^n \alpha$$

- K_S il coefficiente speculare del materiale
- R è il vettore di riflessione
- V è il vettore di vista
- n coefficiente di brillantezza, per adattarsi anche ai riflettori non perfetti, come ad esempio plastica o di cera.



Riflessione speculare

$$E_s = L\, K_s \cos^n\alpha$$

L’angolo α è compreso tra la direzione della luce verso l’osservatore e il vettore di riflessione, più si avvicina allo 0 e più intensa sarà la riflessione speculare per l’osservatore, quindi il termine $\cos^n\alpha$ viene usato per modellare la riduzione di intensità senza particolari legami fisici, ma solo perchè il coseno è facile da calcolare come prodotto di punti e il termine n può essere regolato ad hoc.

Inoltre, anche in questo caso i vettori R e V si possono normalizzare e si può tener conto anche dell’attenuazione dell’intensità dell’illuminazione.

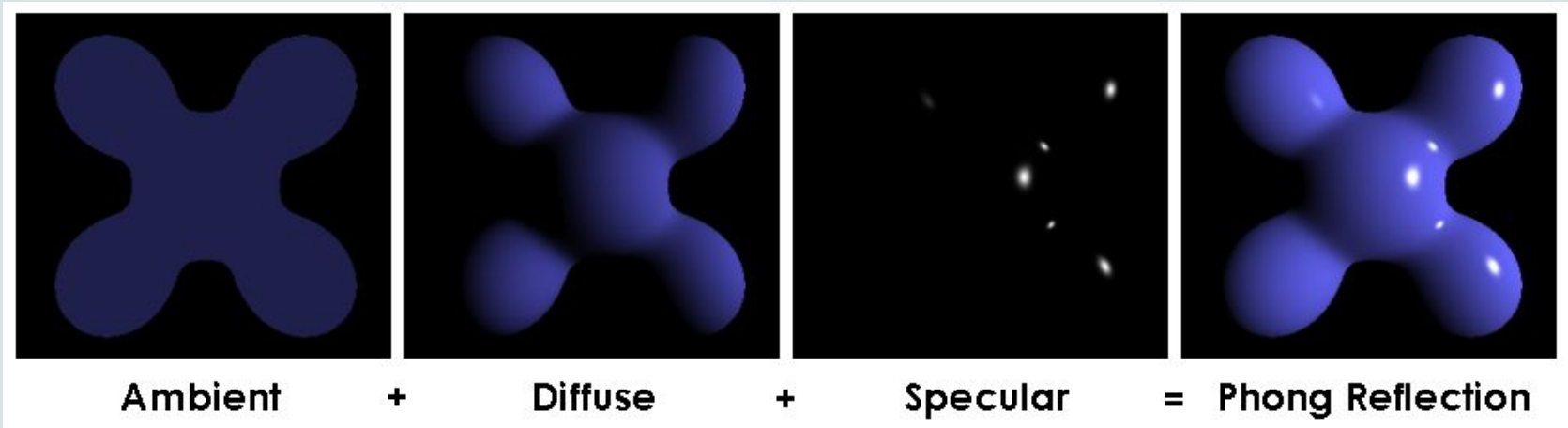
$$E_s = F_{att}\, L\, K_s\, (\overline{R \cdot V})^n]$$

Modello di illuminazione di Phong - Formulazione

Quindi l'illuminazione di una sola sorgente luminosa con un unico componente cromatica è data da:

$$E_{tot} = F_{att}\, L\, [\, K_D\, (L \cdot \overline{N}) + K_S\, (\overline{R \cdot V})^n]$$

Per considerare più sorgenti e varie componenti si utilizza la sovrapposizione degli effetti, quindi si può facilmente definire sottoforma di sommatorie su X, dove X è il numero di fonti luminose.



Modello di illuminazione di Phong - Gestione dei colori

Il modello prevede, inoltre, che la luce venga modellata in un insieme ridotto di lunghezze d’onda, ovvero in un numero ridotto di colori, perchè utilizzare soltanto i colori RGB (e miscele di questi) è coerente con la natura dei monitor;
Ogni componente cromatica viene elaborata, in maniera indipendente dalle altre, tenendo conto con vari coefficienti delle seguenti proprietà:

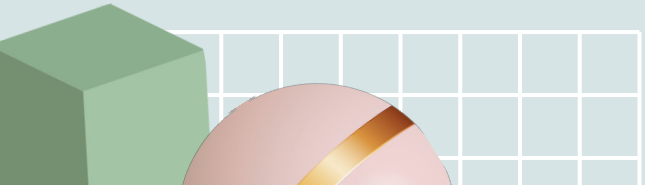
- Proprietà di **riflessione speculare**: controlla la quantità di riflessione speculare e controlla la lucentezza della superficie;
- Proprietà di **riflessione diffusa**: controlla l’intensità relativa della luce riflessa diffusamente;
- Proprietà di **riflessione ambientale**: caratterizza la quantità di luce ambientale riflessa dalla superficie;
- Proprietà **emissive**: l’emissività di una superficie controlla la quantità di luce che essa emette in assenza di luce incidente. E’ importante specificare, però, che la superficie in oggetto non si comporta come una sorgente luminosa per altre superfici ma influisce soltanto sul colore percepito dall’osservatore;

Modello di illuminazione di Phong - Shading

Senza ombreggiatura ogni poligono presente all’interno di un modello geometrico verrebbe renderizzato come un colore solido e costante tale per cui l’immagine in output risulterebbe poco realistica.
Il modello d’illuminazione di Phong consiste in due tipologie di ombreggiatura:

- Shading di **Phong**;
- Shading di **Gouraud**;

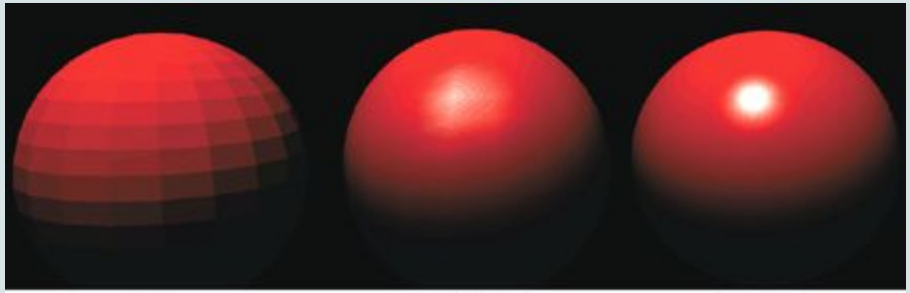
Entrambe le tipologie possono utilizzare l’interpolazione delle linee di scansione, ovvero una tecnica di interpolazione equivalente a quella lineare;



Flat Shading

Il modello più semplice di shading per un poligono è il constant shading (o flat shading), che consiste nell'applicare il modello di illuminazione scelto una sola volta per ogni poligono della scena e poi usare il valore determinato per l'intera superficie del poligono.

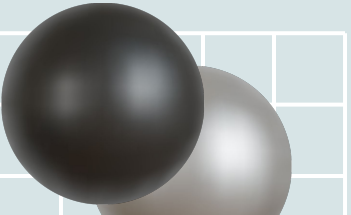
La tecnica di flat shading è estremamente efficiente, poiché l'equazione di illuminazione va calcolata una sola volta per ogni poligono, tuttavia il risultato visivo può non essere del tutto soddisfacente, infatti se la mesh (griglia) di poligoni approssima una superficie più complessa, lascia visibile la suddivisione in poligoni, quindi nell'immagine non si ha una buona resa dell'andamento della superficie.

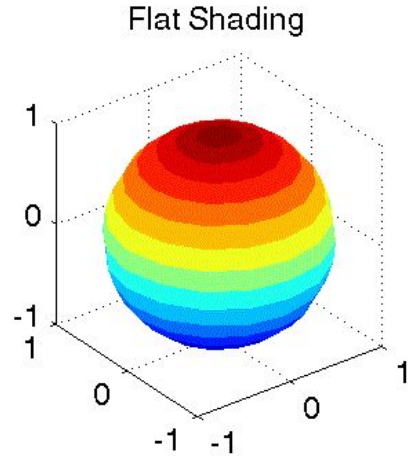


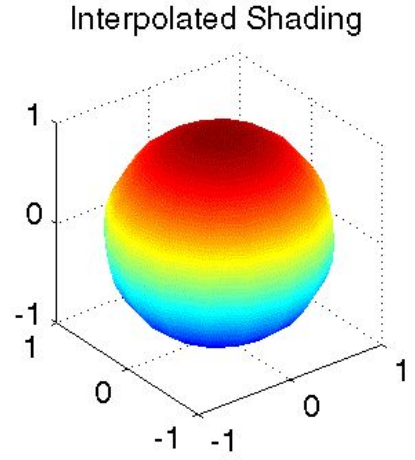
Shading Interpolato

In alternativa a valutare l’equazione di illuminazione in tutti i punti del poligono, si può pensare di calcolarne i valori in ogni vertice ed interpolare linearmente per trovare i valori lungo i lati e nei punti interni.
All’interno di un algoritmo di rasterizzazione applicare l’interpolazione, anche dello shading, comporta uno sforzo minimo.

Questo non risolve però il problema di visualizzare correttamente una superficie curva con una mesh di poligoni, infatti se lo shading (costante o interpolato) viene fatto indipendentemente su ogni poligono si ha una netta visibilità, non voluta, dei bordi tra due poligoni adiacenti causata dalla **brusca variazione** della normale alla superficie.



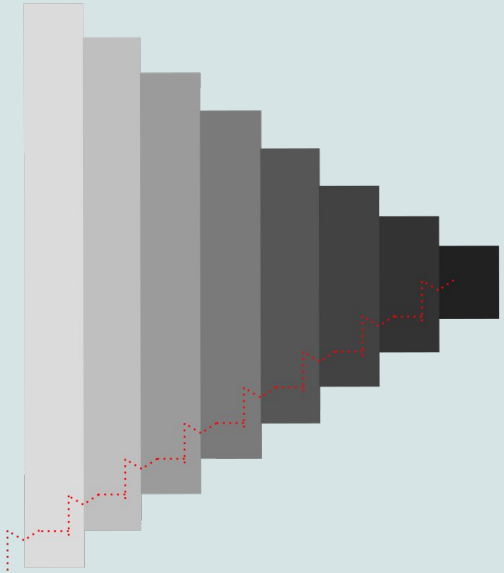




Effetto Mach banding

A causa del cosiddetto effetto Mach banding (dal nome del suo scopritore) anche una maggiore finezza della griglia di poligoni non riduce le discontinuità di shading tra i poligoni adiacenti, infatti questo effetto è quello per cui un oggetto messo vicino ad uno più chiaro risulta più scuro, mentre se messo vicino ad uno più scuro risulta più chiaro.
Questo è causato, fisiologicamente,
dall'effetto inibitore laterale dei **recettori dell'occhio**:
più luce un recettore riceve, più inibisce la risposta
dei recettori vicini.

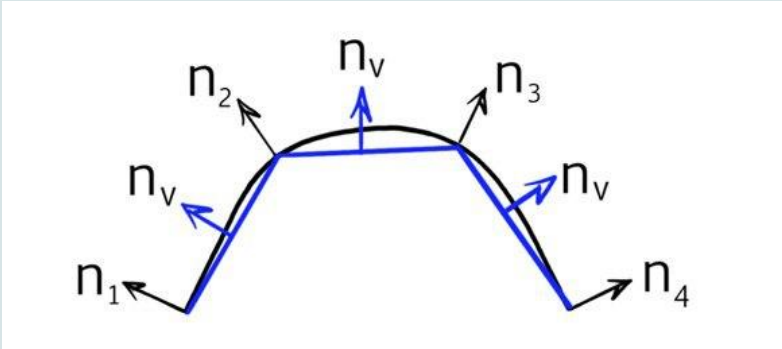
In figura vediamo chiaramente l'effetto:
ogni banda è colorata in modo uniforme, ma
la barra più chiara sembra più luminosa sul bordo in cui si incontra con la barra più scura e viceversa.
In realtà non c'è nessuna transizione !



Gouraud Shading

Per ovviare a questo inconveniente si sono sviluppati dei modelli di shading che tengono conto delle informazioni date da poligoni adiacenti, il modello di Gouraud, è l'evoluzione diretta del metodo con interpolazione del colore su poligoni singoli.

Nel Gouraud shading si tiene conto della geometria effettiva che si sta visualizzando: se la griglia di poligoni rappresenta una superficie curva, per ogni vertice della griglia non si utilizza la normale al poligono ma la normale alla superficie che avrei voluto approssimare.

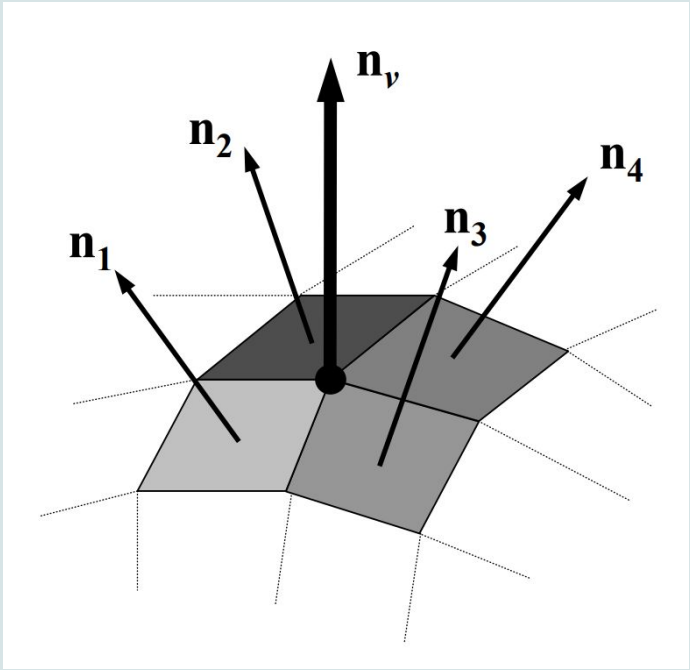


Gouraud Shading

In questa maniera il calcolo dello shading produce lo stesso valore su entrambi i lati dei poligoni che hanno bordi in comune rendendo lo shading complessivo privo di salti.

Però richiede che sia nota la normale alla superficie che si approssima in ogni vertice, se non è disponibile, si approssima con la media pesata delle normali dei poligoni che condividono il vertice.

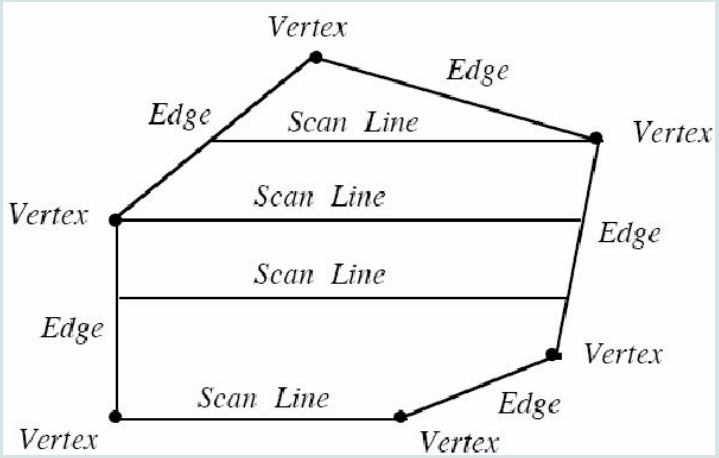
Una volta calcolate (o ricavate analiticamente) le normali in ogni vertice, si applica il modello di illuminazione per calcolare il valore di intensità nel vertice e si interpola linearmente lungo i vertici.



Scan Line

Può essere migliorato utilizzando le **Scan Line**. La scansione di linee è una tecnica di rendering basata sulla suddivisione orizzontale della scena in linee o fasce orizzontali. Per ciascuna scanline, si determinano i punti di intersezione con i bordi dei poligoni presenti nella scena.

Ad esempio, dopo aver effettuato il calcolo dell'illuminazione e dei colori ai vertici tramite Gouraud Shading, è possibile utilizzare la scansione di linee per renderizzare i pixel all'interno dei poligoni in modo più dettagliato.

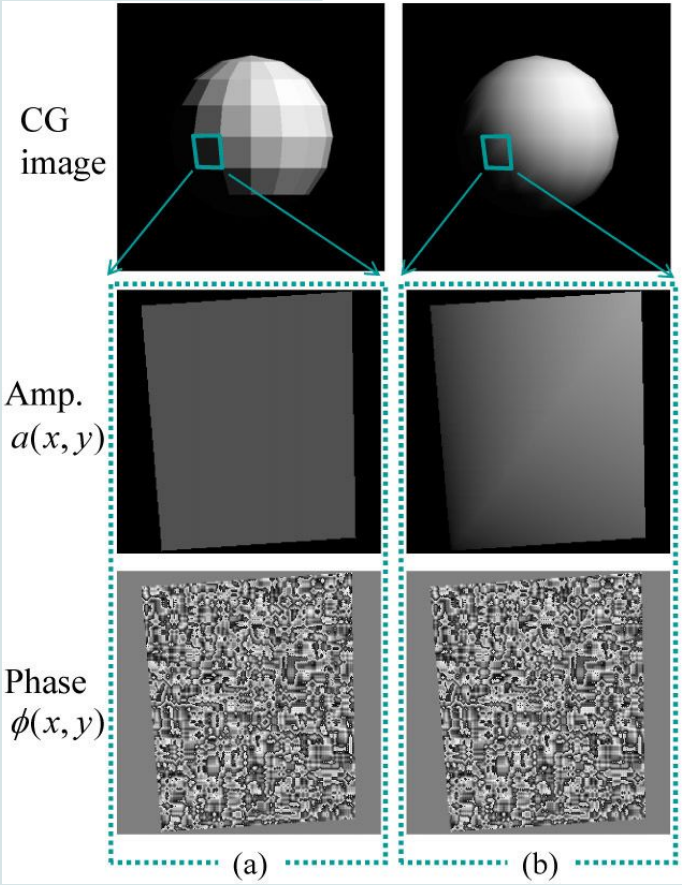


In sintesi, Gouraud Shading e le scanline possono essere usati insieme in un processo di rendering completo per ottenere risultati visivi migliori.

Gouraud Shading

L’interpolazione di Gouraud funziona tendenzialmente bene ed ha il vantaggio di poter essere facilmente implementata, poiché l’equazione di illuminazione va calcolata una sola volta per ogni vertice, ma occorre una **struttura dati** che rappresenti il mesh di poligoni che deve contenere informazioni sui poligoni, vertici, normali e sulle proprietà dei materiali.

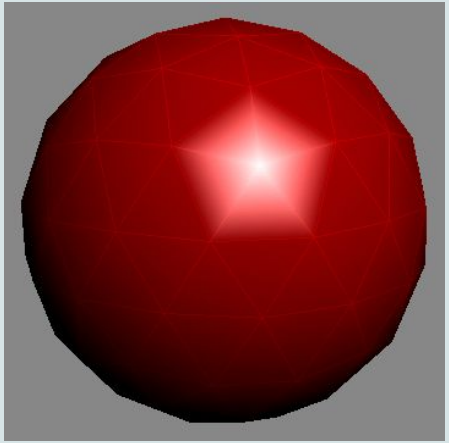
L'approccio Gouraud Shading è spesso utilizzato per migliorare l'aspetto dei modelli 3D in tempo reale, poiché offre un buon compromesso tra qualità visiva e prestazioni computazionali.



Gouraud Shading

Inoltre, essa presenta alcuni svantaggi: Per poligoni di grandi dimensioni, essa potrebbe rivelarsi scarna di diffusione speculare se la parte cade al centro del poligono, ma qualora vi fosse una fonte di luce che punta ad esempio un muro, essa potrebbe essere completamente ignorata, poiché il muro è modellato come un grande poligono allora i quattro vertici potrebbero non essere illuminati dal riflettore;

In conclusione, possiamo osservare che è carente quando la luminosità di una luce speculare dipende fortemente dal modo in cui è centrata su un vertice; Ciò è particolarmente enfatizzato quando gli oggetti o le luci hanno un animazione;



Phong Shading

Il Phong shading è più costoso in termini computazionali rispetto al Gouraud shading, poiché non si interpola le intensità di colore dei vertici, ma si interpolano le normali sui vertici di ciascun poligono per determinare la normale associata a ciascun pixel.
Inoltre, esistono implementazioni in hardware del Gouraud shading che consentono il suo uso in tempo reale, ma ciò non è normalmente vero per il Phong shading.

Attenzione a non confondere Phong model con Phong shading!

Phong Shading

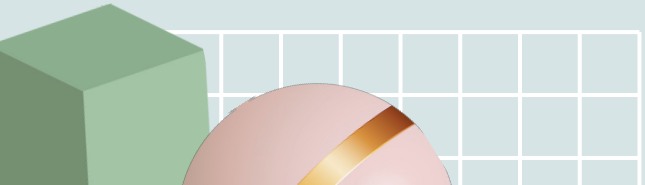
Per prima cosa si determinano le normali nei vertici, anche in questo caso possiamo usare la media pesata delle normali ai poligoni.

Quindi si effettua una interpolazione lineare sulle normali e per ogni punto si applica l'equazione di illuminazione.

Il modo più comune per interpolare le normali è il seguente:

1. Supposti x_0 e x_1 due punti le cui normali sono note che chiameremo n_0 e n_1;
2. Considerato un terzo punto, posto a una frazione ***α*** della congiungente tra x_0 e x_1, allora la normale interpolata in questo punto è pari a:

$$n_\alpha = \frac{(1-\alpha)n_0 + \alpha n_1}{|(1-\alpha)n_0 + \alpha n_1|}$$



Phong Shading

Il grande vantaggio è che non si perdono né le piccole luci speculari né le riflessioni, contrariamente all'approccio di Gouraud, quando essi si trovano all'interno del poligono; Inoltre, la luminosità di una luce speculare non è così sensibile alla posizione in cui essa è centrata.

Il principale svantaggio di questo modello è che tutte le informazioni sulle componenti cromatiche e sulle direzioni delle luci devono necessariamente essere conservate fino alla fase finale del rendering così da poter calcolare l'illuminazione in ogni pixel dell'immagine finale;

Un potenziale problema di entrambe le tecniche di ombreggiatura deriva dal fatto che entrambe eseguono l'interpolazione nelle coordinate dello schermo o della finestra di visualizzazione !
Ciò non riflette correttamente le dimensioni del poligono e implica che, talune volte, l'ombreggiatura conferita ad un poligono sia poco ottimale causando effetti visivi indesiderati.

03

Modello di illuminazione di Cook-Torrance

Modello di illuminazione di Cook-Torrance

Introdotto sulla base del modello d’illuminazione di Bling (1973), il modello di illuminazione di Cook-Torrance (1982) è un modello d’illuminazione alternativo al modello di Phong, in grado di catturare meglio le proprietà di riflessione di una gamma più ampia di materiali.

Rispetto a quello di Phong ha una diversa idea di fondo, il modello utilizza una tecnica basata su **microfaccettature,** avvalendosi delle equazioni di Fresnel per il calcolo dell’intensità di riflessione; Ciò garantisce una migliore gestione delle superfici ruvide e una migliore gestione delle variazioni di riflessione dovute alla presenza di angoli di visione radenti;

La tecnica lavora con l’assunzione secondo cui ogni superficie sia costituita da piccoli pezzi definiti sfaccettature.

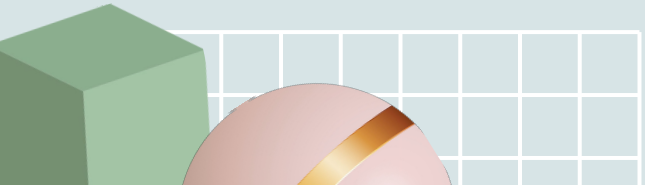
Riflessione Speculare

Similmente al modello di Phong, anche il modello di Cook-Torrance considera la riflessione come formata da componenti ambientali, diffuse e speculari in modo separato;

Le componenti ambientali e diffuse seguono le stesse considerazioni già proposte per il modello di Phong, mentre la componente speculare è definita dalla seguente relazione:

$$E_S = \frac{FDG}{(\vec{N} \cdot \vec{V})\,(\vec{N} \cdot \vec{L})}$$

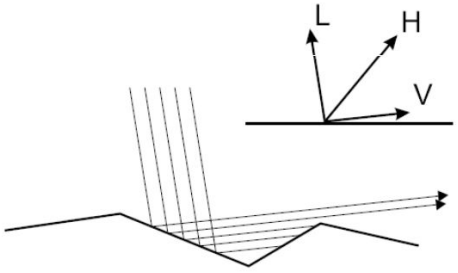
- F è il fattore di Fresnel.
- D dipende dalla distribuzione delle microfacce e modella la rugosità del materiale.
- G è un termine che dipende dalla geometria della superficie.



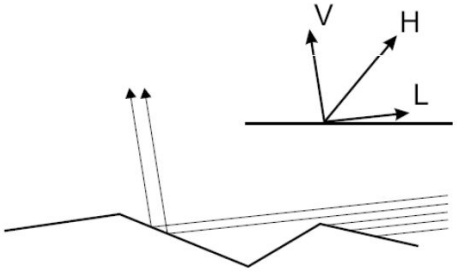
Riflessione Speculare

Il modello di Cook-Torrance è più avanzato e tiene conto della rugosità e della distribuzione delle microfacce, attraverso i fattori F, D e G per il calcolo dell'illuminazione speculare e diffusa; In particolare, possiamo considerare delle funzioni in base alle caratteristiche del materiale:

- Il termine **D = f(L, V)**, modella la rugosità del materiale utilizzando una distribuzione delle microfacce, ad esempio Cook e Torrance utilizzarono la distribuzione di Beckmann-Spizzichino.
- Il fattore **G = f(L, V)** è un fattore di attenuazione che tiene conto degli effetti di masking e shadowing delle microfacce.
- Il fattore **F = f(L, V, λ)** indica la quantità di luce riflessa rispetto a quella incidente, si può notare che dipende anche dalla lunghezza d'onda λ, per cui il colore delle riflessioni varierà al variare di angoli di incidenza e riflessione.



Effetto di Masking



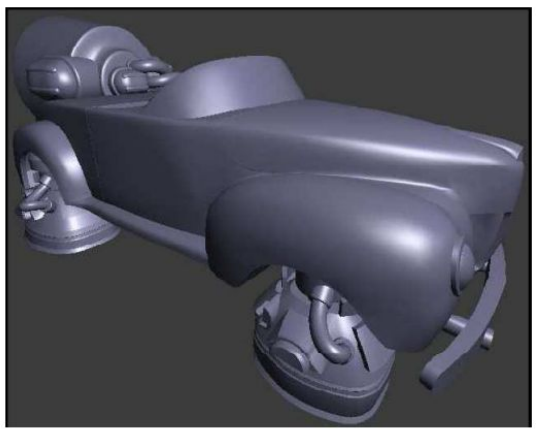
Effetto di Shadowing

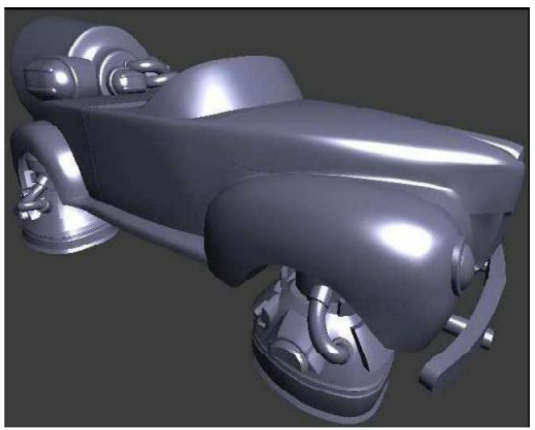
Conservazione dell’energia

Questo modello introduce anche un altro principio fisico, cioè la **conservazione dell’energia,** infatti avremo che la luce incidente deve essere pari alla luce riflessa, quindi, essendo la luce riflessa formato da una componente diffusiva e da una componente speculare avremo:

$$E = E_D + E_S \rightarrow K_S + K_D = 1$$

In seno alle principali differenze individuate tra i due modelli possiamo affermare che il modello di Phong è un approccio più semplice che considera tre componenti illuminazione, invece il modello di Cook-Torrance è più avanzato, ma è possibile applicare varie approssimazioni e semplificazioni per renderlo computazionalmente valido.





Phong

Cook-Torrance

Svantaggi

Tuttavia, come qualsiasi modello, presenta alcuni svantaggi e limitazioni. Ecco alcuni dei principali svantaggi del modello Cook-Torrance:

- **Complessità computazionale**, soprattutto per determinare la distribuzione delle microfacce e il termine G di geometria.
- **Sensibilità ai parametri**, il modello dipende da vari parametri, la scelta e la regolazione di questi parametri possono influenzare significativamente i risultati visivi, e trovare i valori ottimali può richiedere una sperimentazione intensiva.
- **Limitazioni** sulla descrizione delle superfici, poiché assume che tutti i materiali hanno microfacce ruvide

Nonostante questi svantaggi, il modello di Cook-Torrance rimane un potente strumento nella computer grafica e, attualmente, viene spesso utilizzato per simulare il comportamento della luce su superfici ruvide.

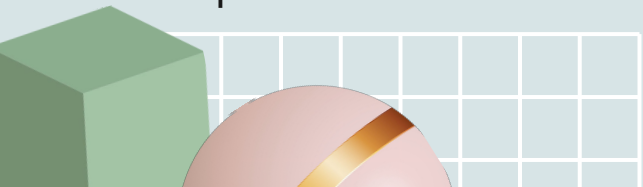
Silicon Graphics

SGI è stata una delle prime aziende a utilizzare i processori MIPS, che erano progettati specificamente per le applicazioni di grafica 3D.

L'azienda ha anche sviluppato una serie di software proprietari per la grafica 3D, tra cui il sistema operativo IRIX e il software di rendering Indigo.

SGI ha avuto un ruolo importante nello sviluppo della grafica 3D moderna. I suoi computer sono stati utilizzati per creare alcuni dei film e dei videogiochi più iconici degli anni '90, tra cui **Jurassic Park, Toy Story e Quake**.

SGI ha cessato le attività nel 2009, ma la sua eredità continua a vivere. I suoi computer e software hanno contribuito a definire l'aspetto della grafica 3D moderna e hanno avuto un impatto duraturo sull'industria dell'intrattenimento.



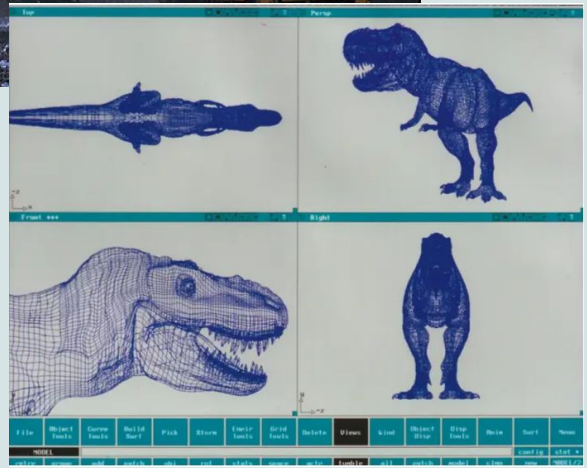
Silicon Graphics

Negli anni '90, a causa delle limitate risorse di elaborazione e delle capacità hardware più limitate rispetto a oggi, spesso si preferivano modelli di illuminazione più **semplici** e computazionalmente meno intensivi. Inoltre, la maggior parte delle applicazioni di grafica 3D dell'epoca si concentrava su rendering in tempo reale per applicazioni come i videogiochi.

Tuttavia, con l'avanzare della tecnologia hardware e dei software di grafica 3D, il modello di Cook-Torrance e altri modelli avanzati sono diventati più accessibili e popolari negli anni successivi.

Infatti i computer Silicon Graphics del tempo utilizzavano spesso una combinazione di modelli e tecniche di illuminazione per ottenere i migliori risultati possibili. Ad esempio, potevano utilizzare il modello di Gouraud per calcolare l'illuminazione iniziale e utilizzare l'illuminazione di Phong o la radiosità per migliorare la precisione.

Alcune Creazioni - Jurassic Park



Per questa scena ci sono volute circa 4 mesi di lavoro, tra realizzazione del modello, studio dell’illuminazione e applicazioni dei modelli !

"Jurassic Park," diretto da Steven Spielberg e rilasciato nel 1993, è spesso considerato uno dei primi film a convincere il pubblico sull'efficacia e il potenziale della grafica 3D nei film.

Alcuni punti chiave che hanno reso "Jurassic Park" un pioniere nella grafica 3D cinematografica:

- **Dinosauri Realistici**, Il film presentava dinosauri realistici e convincenti generati al computer.
- I**ntegrazione Visiva**: è stato uno dei primi film a dimostrare con successo l'integrazione visiva tra attori dal vivo e creature generate al computer, contribuendo a rendere la narrazione più coinvolgente e realistica.

Alcune Creazioni - Toy Story

"Toy Story," rilasciato nel 1995 e diretto da John Lasseter, è è stato il primo lungometraggio animato completamente al computer.
Il film ha dimostrato che la CGI poteva essere utilizzata per creare personaggi e ambienti realistici e dettagliati. I personaggi erano caratterizzati da un'animazione fluida e realistica, mentre gli ambienti, come la casa di Andy e il parco giochi, erano accuratamente riprodotti, creando un senso di immersione totale nello spettatore.

La produzione parte nel 1991, I primi due anni di lavorazione sono stati dedicati alla ricerca e allo sviluppo di nuove tecniche di animazione.
La vera e propria realizzazione è iniziata nel 1993. Un team di circa 100 animatori ha lavorato al progetto per due anni, creando oltre 90.000 fotogrammi.

04

Implementazione in Matlab

Implementazione in MatLab

- La simulazione in MatLab è stata realizzata a partire dall'implementazione di due *script:*
  - *PhongModel_main*
  - *PhongModel_function*

Implementazione in MatLab

- **PhongModel_main**:

coloreLuce: è un array che consente di definire la tripla [R,G,B]. Esso è stato impostato secondo i seguenti valori: [1 1 1]. Tale configurazione consente di inserire una luce bianca che contiene le componenti cromatiche in egual quantità. Configurare la luce in questo modo consente, infatti, di valutare l'effetto del materiale sulla riflessione e diffusione della luce senza alcuna forma di alterazione o distorsione provocata da particolari configurazioni cromatiche. Ciò è di fondamentale importanza poiché l'obiettivo della simulazione è osservare come il materiale reagisce in termini di riflessione ambientale, diffusa e speculare indipendentemente dal colore della luce che lo colpisce;

```
% Luce bianca
coloreLuce =[1 1 1];

tipoSuperfice = 'metal';
ka = 0.1;
kd = 0.1;
ks = 1.0;
ke = 5.0;
scr = 0.5;


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Metal');

tipoSuperfice = 'shiny';
ka = 0.1;
kd = 0.6;
ks = 0.7;
ke = 5.0;
scr = 1.0;

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Shiny');

tipoSuperfice = 'diffuse';
ka = 0.1;
kd = 0.7;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('diffuse');

tipoSuperfice = 'ambient';
ka = 1.0;
kd = 0.0;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('ambient');
```

Implementazione in MatLab

- **PhongModel_main**:

tipoSuperficie: il seguente parametro indica il tipo di superficie sotto analisi. Per la simulazione sono state individuate quattro tipologie di superficie:

- ***Metal:*** rappresenta una superficie di tipo metallico gode di un'alta riflessione speculare ed un alto parametro di emissività;
- ***Shiny:*** rappresenta una superficie di tipo brillante. Tale superficie ha un comportamento simile a quella metallica ma con coefficiente di riflessione speculare ed emissività più moderati rispetto a quest'ultima;
- ***Diffuse:*** rappresenta una superficie in grado di diffondere uniformemente la luce in tutte le direzioni. Questo comportamento è modellato dalle seguenti posizioni:
  - Assenza di riflessione speculare;
  - Alta riflessione diffusa;
- ***Ambient:*** rappresenta una superficie in grado di riflettere la luce ambientale in modo uniforme in tutte le direzioni. Tale superficie prevede che non vi siano né componenti di riflessione speculare né di riflessione diffusa per cui ambo i coefficienti sono nulli;

```
% Luce bianca
coloreLuce =[1 1 1];

tipoSuperfice = 'metal';
ka = 0.1;
kd = 0.1;
ks = 1.0;
ke = 5.0;
scr = 0.5;


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Metal');

tipoSuperfice = 'shiny';
ka = 0.1;
kd = 0.6;
ks = 0.7;
ke = 5.0;
scr = 1.0;

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Shiny');

tipoSuperfice = 'diffuse';
ka = 0.1;
kd = 0.7;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('diffuse');

tipoSuperfice = 'ambient';
ka = 1.0;
kd = 0.0;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('ambient');
```

Implementazione in MatLab

- **PhongModel_main**:

Coefficienti della relazione di Phong:

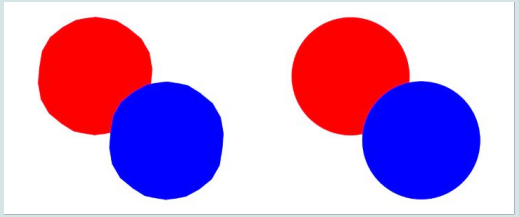
- ***Coefficiente di riflessione ambientale*** k_a ***:*** controlla quanto la superficie riflesse la luce ambientale. Maggiore è il suo valore e maggiore sarà l'effetto di riflessione ambientale percepito. L'insieme dei valori che questo coefficiente può assumere è il seguente:
- ***Coefficiente di riflessione diffusa*** k_d ***:*** controlla quanto la superficie diffonde la luce proveniente dalla sorgente luminosa. In termini di proporzionalità vale quanto detto per il coefficiente k_a .E' importante osservare che la riflessione avviene in maniera uniforme in tutte le direzioni. L'insieme dei valori che questo coefficiente può assumere è:
- ***Coefficiente di riflessione speculare*** k_s ***:*** controlla quanto la superficie riflette la luce in maniera speculare e quanto il suo comportamento si avvicini a quello di uno specchio. L'insieme dei valori che questo coefficiente può assumere è:
- ***Coefficiente di emissività*** k_e ***:*** controlla la quantità di luce propria che la superficie riesce ad emettere. Questo coefficiente consente di quantificare la capacità della superficie di irradiare luce indipendentemente dalla quantità di luce incidente. Maggiore sarà il suo valore e più intensa sarà l'emissione luminosa. I valori che può assumere questo coefficiente sono:
- ***Luce riflessa*** (scr)***:*** rappresenta la combinazione di illuminare e colore della superficie. Viene utilizzata per calcolare la luce riflessa da essa.

```
% Luce bianca
coloreLuce =[1 1 1];

tipoSuperfice = 'metal';
ka = 0.1;
kd = 0.1;
ks = 1.0;
ke = 5.0;
scr = 0.5;


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Metal');

tipoSuperfice = 'shiny';
ka = 0.1;
kd = 0.6;
ks = 0.7;
ke = 5.0;
scr = 1.0;

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('Shiny');

tipoSuperfice = 'diffuse';
ka = 0.1;
kd = 0.7;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti


PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('diffuse');

tipoSuperfice = 'ambient';
ka = 1.0;
kd = 0.0;
ks = 0.0;
ke = 1.0;
scr = 1.0;
% Con Ks = 0 i valori esatti di Ke e Scr non sono rilevanti

PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr);
title('ambient');
```

Implementazione in MatLab

- **PhongModel_function**:

[x_Sfera, y_Sfera, z_Sfera]=sphere(180): il seguente comando consente la creazione di una griglia di punti che rappresentano una sfera di tipo tridimensionale. La funzione sphere() genera, automaticamente, una sfera di raggio unitario con risoluzione, in questo caso, pari a 180.
Il valore della risoluzione, è doveroso specificarlo, indica il numero di punti che verrà utilizzato per la rappresentazione della figura. Minore sarà la risoluzione passata in input e più la sfera creata risulterà poligonale. Nella Figura sottostante è proposto un confronto tra una coppia di sfere generata con una risoluzione pari a 10 ed una coppia generata con una risoluzione pari a 180.



```
% tipoSuperficie: una stringa che indica il tipo di superficie da
% illuminare;
% coloreluce: il colore della luce utilizzata per illuminare le superficie;
% I valori Ka, Kd, Ks, Ke e Scr rappresentano i coefficienti utilizzati nel
% modello di Phong;
% Ka: Coeffiente di riflessione ambientale;
% Kd: Coeffiente di riflessione diffusiva;
% Ks: Coeffiente di riflessione speculare ambiente;
% Ke: Coeffiente di emissività;
% Src: Luce riflessa ossia combinazione tra illuminazione e il colore della
% superficie;

%------------------------------CODICE ----------------------------------

function PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr)

figure;

% La funzione sphere() genera, automaticamente, una sfera di raggio
% unitario con risoluzione, in questo caso, pari a 180.

[x_Sfera, y_Sfera, z_Sfera] = sphere(180);

% Consente di disegnare una superficie sferica tridimensionale a partire
% dai punti ottenuti dalla funzione sphere()

handle_Sfera1 = surf(x_Sfera, y_Sfera, z_Sfera);
hold on;

% Sfera 2 leggermente spostata

handle_Sfera2 = surf(x_Sfera-2, y_Sfera-2, z_Sfera);

% La funzione light crea una sorgente di luce
% In questo caso ne sto creando 2 in posizioni diverse
%
posizione_L1 = light('Position', [1 -1 1], 'Color', coloreLuce);
posizione_L2 = light('Position', [-3 0 3], 'Color', coloreLuce);


set(handle_Sfera1, 'FaceLighting', 'phong', 'FaceColor', 'b', 'EdgeColor', 'none'
set(handle_Sfera2, 'FaceLighting', 'phong', 'FaceColor', 'r', 'EdgeColor', 'none'
```

Implementazione in MatLab

- **PhongModel_function**:

handle_SferaX=surf(x,y,z): suddetta funzione, nella sua generalità, consente di disegnare una superficie sferica tridimensionale a partire dai punti ottenuti dalla funzione sphere(). Particolarizziamo suddetta riga di codice per entrambe le sfere disegnate:

- ***Sfera1: per la prima sfera la funzione surf prevede in ingresso i seguenti parametri (x_Sfera, y_Sfera, z_Sfera) ovvero le coordinate x, y e z ottenute dalla funzione sphere();***
- ***Sfera2: la seconda sfera, invece, è stata volutamente disegnata leggermente spostata rispetto alla prima; per tal motivo, dunque, alla funzione surf vengono passati i seguenti parametri (x_Sfera-2, y_Sfera+2, z_Sfera) per cui essa risulterà essere traslata di -2 lungo l'orizzontale e +2 lungo la verticale***

```
% tipoSuperficie: una stringa che indica il tipo di superficie da
% illuminare;
% coloreluce: il colore della luce utilizzata per illuminare le superficie;
% I valori Ka, Kd, Ks, Ke e Scr rappresentano i coefficienti utilizzati nel
% modello di Phong;
% Ka: Coeffiente di riflessione ambientale;
% Kd: Coeffiente di riflessione diffusiva;
% Ks: Coeffiente di riflessione speculare ambiente;
% Ke: Coeffiente di emissività;
% Src: Luce riflessa ossia combinazione tra illuminazione e il colore della
% superficie;

%------------------------------CODICE ----------------------------------

function PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr)

figure;

% La funzione sphere() genera, automaticamente, una sfera di raggio
% unitario con risoluzione, in questo caso, pari a 180.

[x_Sfera, y_Sfera, z_Sfera] = sphere(180);

% Consente di disegnare una superficie sferica tridimensionale a partire
% dai punti ottenuti dalla funzione sphere()

handle_Sfera1 = surf(x_Sfera, y_Sfera, z_Sfera);
hold on;

% Sfera 2 leggermente spostata

handle_Sfera2 = surf(x_Sfera-2, y_Sfera-2, z_Sfera);

% La funzione light crea una sorgente di luce
% In questo caso ne sto creando 2 in posizioni diverse
%
posizione_L1 = light('Position', [1 -1 1], 'Color', coloreLuce);
posizione_L2 = light('Position', [-3 0 3], 'Color', coloreLuce);


set(handle_Sfera1, 'FaceLighting', 'phong', 'FaceColor', 'b', 'EdgeColor', 'none'
set(handle_Sfera2, 'FaceLighting', 'phong', 'FaceColor', 'r', 'EdgeColor', 'none'
```

Implementazione in MatLab

- **PhongModel_function**:

```
% tipoSuperficie: una stringa che indica il tipo di superficie da
% illuminare;
% coloreluce: il colore della luce utilizzata per illuminare le superficie;
% I valori Ka, Kd, Ks, Ke e Scr rappresentano i coefficienti utilizzati nel
% modello di Phong;
% Ka: Coeffiente di riflessione ambientale;
% Kd: Coeffiente di riflessione diffusiva;
% Ks: Coeffiente di riflessione speculare ambiente;
% Ke: Coeffiente di emissività;
% Src: Luce riflessa ossia combinazione tra illuminazione e il colore della
% superficie;

%------------------------------CODICE ----------------------------------

function PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr)

figure;

% La funzione sphere() genera, automaticamente, una sfera di raggio
% unitario con risoluzione, in questo caso, pari a 180.

[x_Sfera, y_Sfera, z_Sfera] = sphere(180);

% Consente di disegnare una superficie sferica tridimensionale a partire
% dai punti ottenuti dalla funzione sphere()

handle_Sfera1 = surf(x_Sfera, y_Sfera, z_Sfera);
hold on;

% Sfera 2 leggermente spostata

handle_Sfera2 = surf(x_Sfera-2, y_Sfera-2, z_Sfera);

% La funzione light crea una sorgente di luce
% In questo caso ne sto creando 2 in posizioni diverse
%
posizione_L1 = light('Position', [1 -1 1], 'Color', coloreLuce);
posizione_L2 = light('Position', [-3 0 3], 'Color', coloreLuce);


set(handle_Sfera1, 'FaceLighting', 'phong', 'FaceColor', 'b', 'EdgeColor', 'none'
set(handle_Sfera2, 'FaceLighting', 'phong', 'FaceColor', 'r', 'EdgeColor', 'none'
```

setSfera1:

- ***('FaceLighting', 'phong'): la seguente coppia di valori specifica quale algoritmo di illuminazione utilizzare; in questo caso, naturalmente, è stato scelto il modello d'illuminazione di Phong;***
- ***('FaceColor', 'b'): la seguente coppia viene utilizzata per specificare il colore della prima sfera che, in questo caso, sarà blu;***
- ***('EdgeColor', 'none'): specifica che la sfera disegnata non avrà alcun bordo intorno alla propria superficie;***
- ***('AmbientStrength', k_a :(specifica l'intensità della riflessione ambientale della superficie con dipendenza dal parametro ka;***
- ***('DiffuseStrength', k_d :(specifica l'intensità della riflessione diffusa dalla superficie con dipendenza dal parametro kd;***

Implementazione in MatLab

- **PhongModel_function**:

setSfera1:

- ***('SpecularStrength', k_s :(specifica l'intensità della riflessione speculare della superficie con dipendenza dal valore ks;***
- ***('SpecularExponent', k_e :(specifica il coefficiente di emissività della superficie ke;***
- ***('SpecularColorReflectance', scr): specifica la quantità di luce riflessa dalla superficie in relazione al valore scr specificato;***
- ***('BackFaceLighting', 'unlit'): attraverso questa coppia, infine, viene specificato che l'illuminazione deve essere applicata soltanto alla figura rappresentata. Ciò garantisce, dunque, che il retro della superficie rimanga non illuminato;***

```
% tipoSuperficie: una stringa che indica il tipo di superficie da
% illuminare;
% coloreluce: il colore della luce utilizzata per illuminare le superficie;
% I valori Ka, Kd, Ks, Ke e Scr rappresentano i coefficienti utilizzati nel
% modello di Phong;
% Ka: Coeffiente di riflessione ambientale;
% Kd: Coeffiente di riflessione diffusiva;
% Ks: Coeffiente di riflessione speculare ambiente;
% Ke: Coeffiente di emissività;
% Src: Luce riflessa ossia combinazione tra illuminazione e il colore della
% superficie;

%------------------------------CODICE ----------------------------------

function PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr)

figure;

% La funzione sphere() genera, automaticamente, una sfera di raggio
% unitario con risoluzione, in questo caso, pari a 180.

[x_Sfera, y_Sfera, z_Sfera] = sphere(180);

% Consente di disegnare una superficie sferica tridimensionale a partire
% dai punti ottenuti dalla funzione sphere()

handle_Sfera1 = surf(x_Sfera, y_Sfera, z_Sfera);
hold on;

% Sfera 2 leggermente spostata

handle_Sfera2 = surf(x_Sfera-2, y_Sfera-2, z_Sfera);

% La funzione light crea una sorgente di luce
% In questo caso ne sto creando 2 in posizioni diverse
%
posizione_L1 = light('Position', [1 -1 1], 'Color', coloreLuce);
posizione_L2 = light('Position', [-3 0 3], 'Color', coloreLuce);


set(handle_Sfera1, 'FaceLighting', 'phong', 'FaceColor', 'b', 'EdgeColor', 'none'
set(handle_Sfera2, 'FaceLighting', 'phong', 'FaceColor', 'r', 'EdgeColor', 'none'
```

Implementazione in MatLab

- **PhongModel_function**:

setSfera2: al fine di evitare ridondanze, riporteremo soltanto gli unici parametri con diversa configurazione rispetto alla prima sfera:

- ***('FaceColor', r): la seconda sfera, in questo modo, sarà colorata di rosso;***
- ***('BackFaceLighting', 'unlit'): la seconda sfera, in questo caso, prevede un'illuminazione retrostante***

```
% tipoSuperficie: una stringa che indica il tipo di superficie da
% illuminare;
% coloreluce: il colore della luce utilizzata per illuminare le superficie;
% I valori Ka, Kd, Ks, Ke e Scr rappresentano i coefficienti utilizzati nel
% modello di Phong;
% Ka: Coeffiente di riflessione ambientale;
% Kd: Coeffiente di riflessione diffusiva;
% Ks: Coeffiente di riflessione speculare ambiente;
% Ke: Coeffiente di emissività;
% Src: Luce riflessa ossia combinazione tra illuminazione e il colore della
% superficie;

%------------------------------CODICE ----------------------------------

function PhongModel_function(tipoSuperfice, coloreLuce, ka, kd, ks, ke, scr)

figure;

% La funzione sphere() genera, automaticamente, una sfera di raggio
% unitario con risoluzione, in questo caso, pari a 180.

[x_Sfera, y_Sfera, z_Sfera] = sphere(180);

% Consente di disegnare una superficie sferica tridimensionale a partire
% dai punti ottenuti dalla funzione sphere()

handle_Sfera1 = surf(x_Sfera, y_Sfera, z_Sfera);
hold on;

% Sfera 2 leggermente spostata

handle_Sfera2 = surf(x_Sfera-2, y_Sfera-2, z_Sfera);

% La funzione light crea una sorgente di luce
% In questo caso ne sto creando 2 in posizioni diverse
%
posizione_L1 = light('Position', [1 -1 1], 'Color', coloreLuce);
posizione_L2 = light('Position', [-3 0 3], 'Color', coloreLuce);


set(handle_Sfera1, 'FaceLighting', 'phong', 'FaceColor', 'b', 'EdgeColor', 'none'
set(handle_Sfera2, 'FaceLighting', 'phong', 'FaceColor', 'r', 'EdgeColor', 'none'
```

Implementazione in MatLab

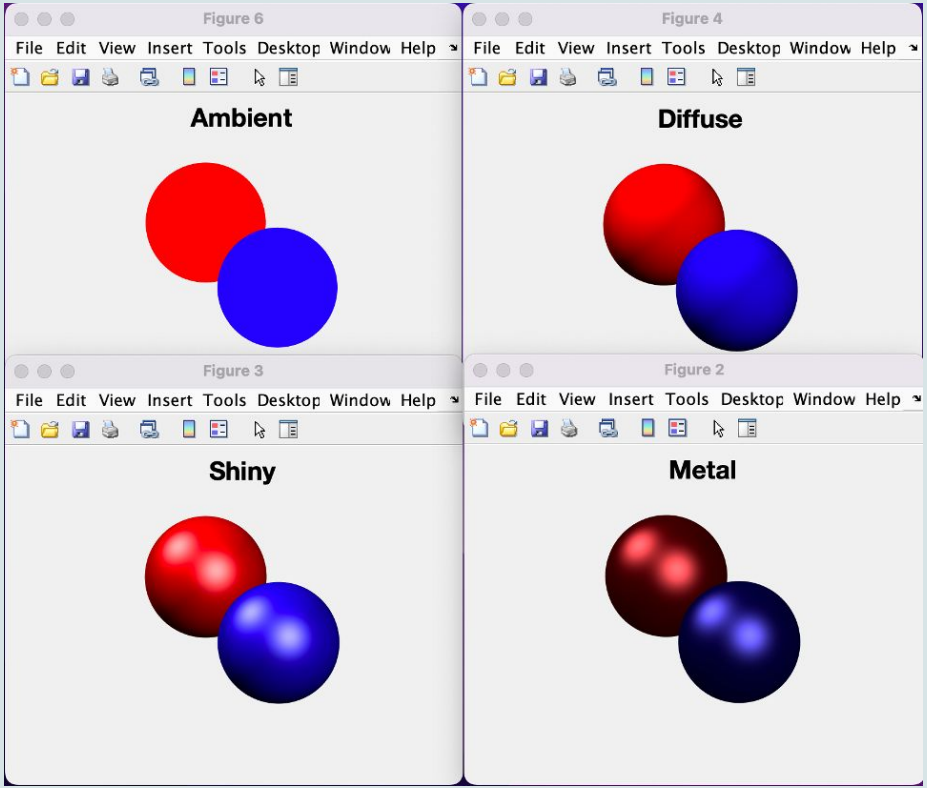
Risultati della Simulazione

- Vengono riportati, di seguito, i valori assegnati ai parametri a fini simulativi:

	Coefficiente di riflessione ambientale (ka)	Coefficiente di riflessione diffusa (kd)	Coefficiente di riflessione speculare (ks):	Coefficiente di emissività (ke):	Luce riflessa (scr):
Ambient	1.0	0.0	0.0	1.0	1.0
Diffuse	0.1	0.7	0.0	1.0	1.0
Shiny	0.1	0.6	0.7	5.0	1.0
Metal	0.1	0.1	1.0	5.0	0.5

Implementazione in MatLab

Risultati della Simulazione



	Coefficiente di riflessione ambientale (ka)	Coefficiente di riflessione diffusa (kd)	Coefficiente di riflessione speculare (ks):	Coefficiente di emissività (ke):	Luce riflessa (scr):
Ambient	1.0	0.0	0.0	1.0	1.0
Diffuse	0.1	0.7	0.0	1.0	1.0
Shiny	0.1	0.6	0.7	5.0	1.0
Metal	0.1	0.1	1.0	5.0	0.5