

Capitolo 6

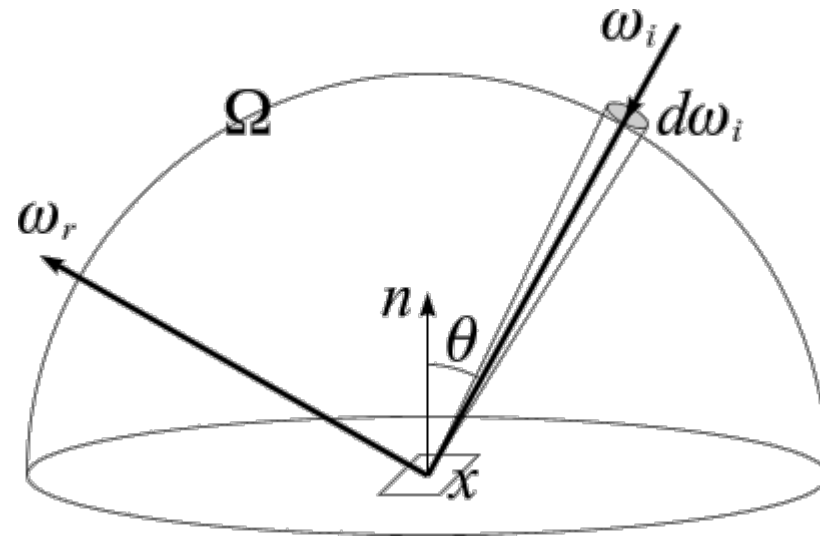
Interazione luce-materia

Diapositive a corredo del libro: "Fondamenti di Grafica Tridimensionale Interattiva"
R. Scateni, P. Cignoni, C. Montani e R. Scopigno - McGrawHill Italia

Modelli di illuminazione

- ❖ **Modello di illuminazione:** formulazione matematica dell'equazione del trasporto dell'energia luminosa
- ❖ L'equazione che risolve questo problema: **equazione di illuminazione**
- ❖ **Lighting:** calcolo del bilancio luminoso
- ❖ **Shading:** calcolo del colore di ogni pixel dell'immagine

L'equazione della radianza



$$L_o(x, \vec{\omega}_r) = L_e(x, \vec{\omega}_r) + L_r(x, \Omega)$$

$$L_o(x, \vec{\omega}_r) = L_e(x, \vec{\omega}_r) + \int_{\Omega} f_r(x, \vec{\omega}_i, \vec{\omega}_r) L_i(x, \vec{\omega}_i) (\vec{\omega}_i \cdot \vec{n}) d\vec{\omega}_i$$

L'equazione della radianza

$$L_o(x, \vec{\omega}_r) = L_e(x, \vec{\omega}_r) + L_r(x, \Omega)$$

- ❖ La luce visibile in un punto della scena è data dalla somma della luce riflessa più la luce emessa

$$L_o(x, \vec{\omega}_r) = L_e(x, \vec{\omega}_r) + \int_{\Omega} f_r(x, \vec{\omega}_i, \vec{\omega}_r) L_i(x, \vec{\omega}_i) (\vec{\omega}_i \cdot \vec{n}) d\vec{\omega}_i$$

- ❖ La luce riflessa è un integrale
- ❖ Somma i contributi di tutte le sorgenti luminose presenti nella scena e tiene conto dell'angolo di riflessione

L'equazione della radianza: parametri

- x punto sulla superficie in cui si calcola l'equazione;
- $\vec{\omega}_r$ direzione che unisce il punto alla posizione dell'osservatore
- $\vec{\omega}_i$ direzione da cui proviene il raggio incidente
- f_r funzione che determina la frazione riflessa di luce incidente
- $\vec{\omega}_i \cdot \vec{n}$ coseno dell'angolo di incidenza rispetto alla normale alla superficie

L'equazione della radianza

- ❖ Calcolo esatto dell'equazione della radianza: operazione complessa e molto costosa
- ❖ Sistema di grafica interattiva: formula utilizzabile per tutti i punti della scena più volte al secondo
- ❖ Semplificazione dell'equazione

Il modello di Phong

- ❖ Modello dovuto a Phong Bui-Tran, prima metà degli anni '70
- ❖ Semplifica lo schema fisico di interazione:
 - ❖ Solo sorgenti puntiformi
 - ❖ No inter-riflessioni
 - ❖ Calcolo locale dell'equazione di illuminazione
 - ❖ Approssimazione con due costanti della funzione di riflessione

Il modello di Phong

- ❖ Simula il comportamento di materiali opachi
- ❖ Non modella la rifrazione: no materiali trasparenti o semi-trasparenti
- ❖ Visione complessiva dominata dall'animazione
- ❖ Non adatto per immagini singole (necessitano di modelli globali)

Il modello di Phong

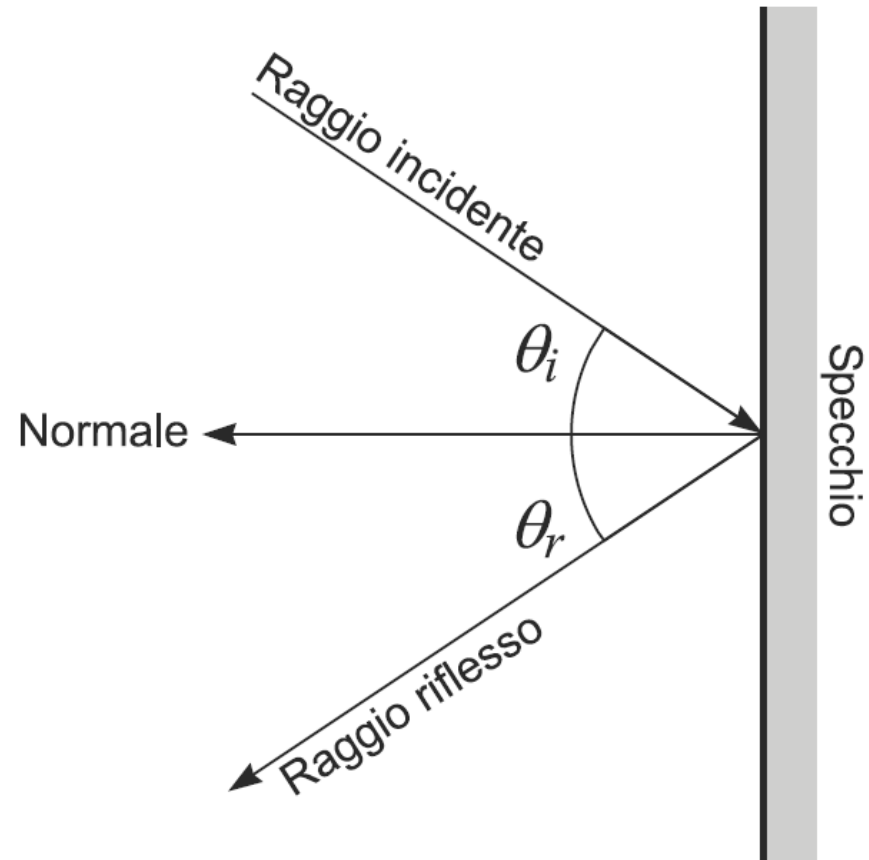
- ❖ Le formule che seguono *banda cromatica*: produzione di un'immagine a diversi livelli di intensità (toni di grigio) piuttosto che diversi colori
- ❖ Quando si utilizza una rappresentazione a colori RGB l'equazione viene calcolata in modo indipendente per ciascuna delle tre componenti cromatiche

Riflessione speculare e diffusa (Phong)

- ❖ **Obiettivo:** approssimare il termine f_r dell'equazione della radianza
- ❖ **Metodo:** semplificazione del fenomeno della riflessione usando le leggi della fisica che regolano la riflessione speculare (Fresnel) e la riflessione diffusa (Lambert)

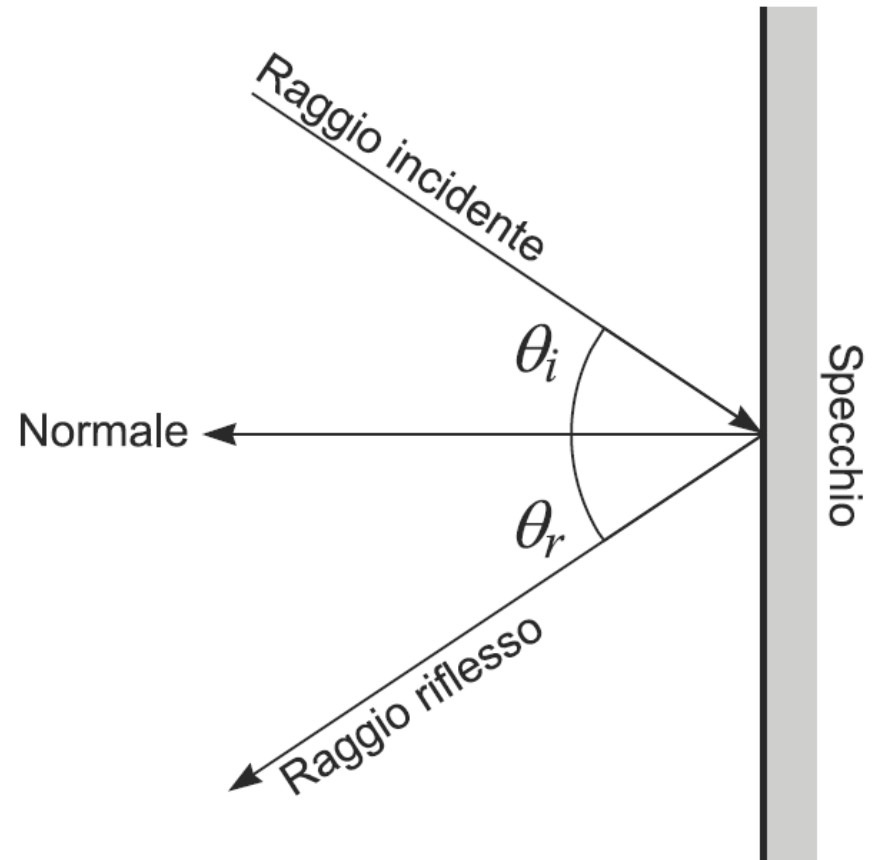
Legge di Fresnel

- ❖ Quando un raggio di luce passa da un mezzo ad un altro con diverso indice di rifrazione raggiunta la superficie di separazione parte del raggio viene riflessa e parte trasmessa
- ❖ La somma delle energie dei due raggi è uguale all'energia del raggio originale



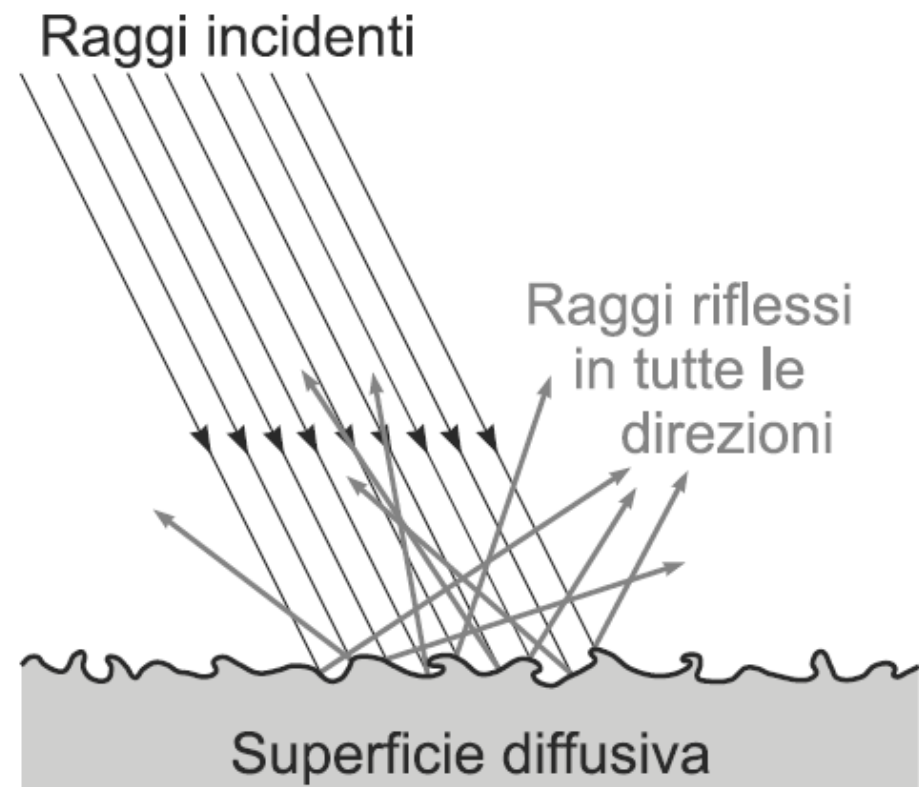
Legge di Fresnel

- ❖ Se da aria a corpo solido non c'è rifrazione si ha solo riflessione
- ❖ L'angolo di incidenza è uguale all'angolo di riflessione
- ❖ Vale per materiali molto lisci e lucidi



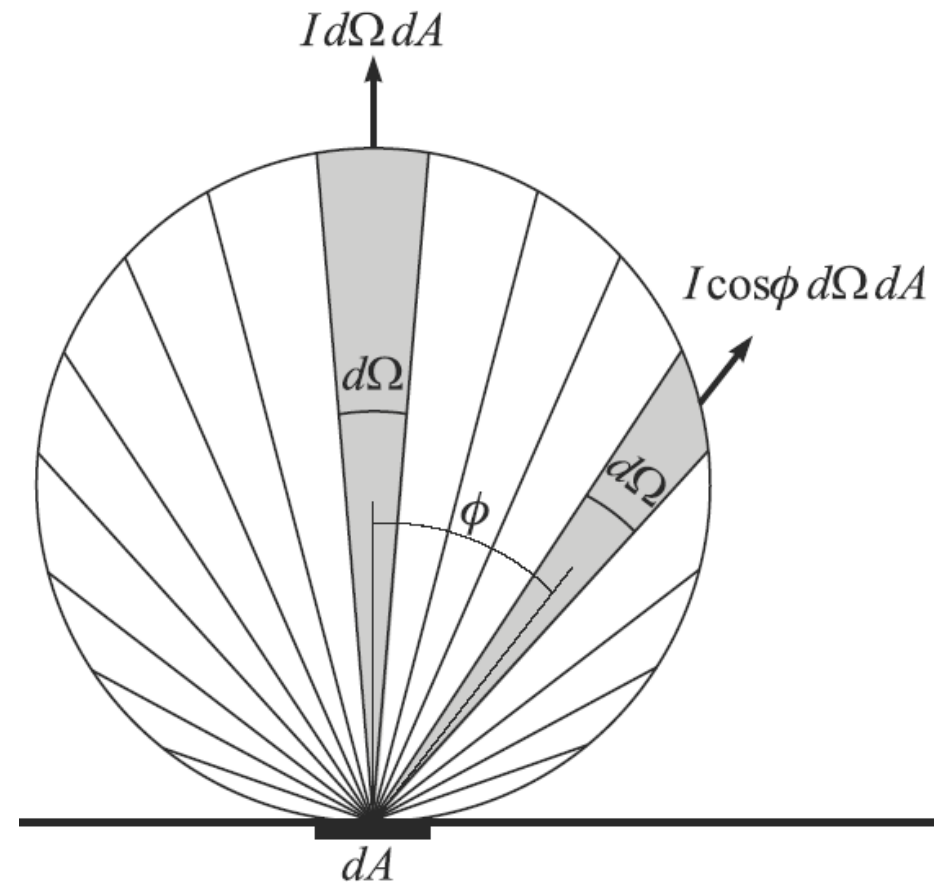
Legge di Lambert

- ❖ Materiali molto opachi (es. gesso e legno) hanno una superficie che, a livello microscopico, ha piccole sfaccettature che riflettono la luce in una direzione casuale



Legge di Lambert

- ❖ Integrando su scala macroscopica: la luce si riflette uniformemente verso tutte le direzioni, con intensità proporzionale al rapporto tra la direzione del raggio incidente e la normale alla superficie in quel punto



Modellazione della riflessione diffusiva

- ❖ Sorgenti luminose puntiformi:

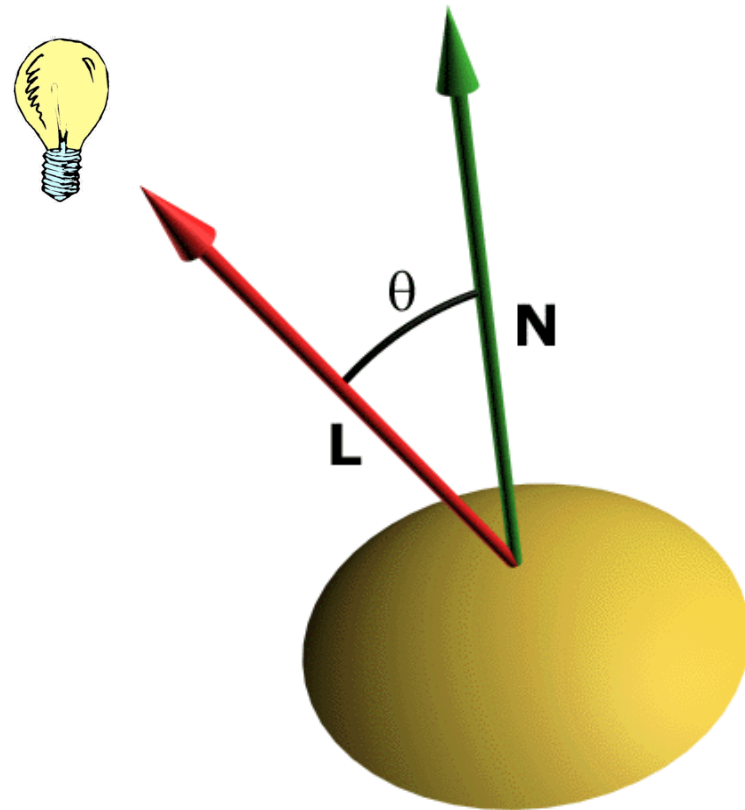
- ❖ posizione nella scena
- ❖ intensità della luce emessa

- ❖ Per calcolare in P con normale \vec{N} :

$$\vec{L} = |L - P|$$

- ❖ Dipendenza solo da θ

$$\cos(\theta) = \vec{L} \cdot \vec{N}$$



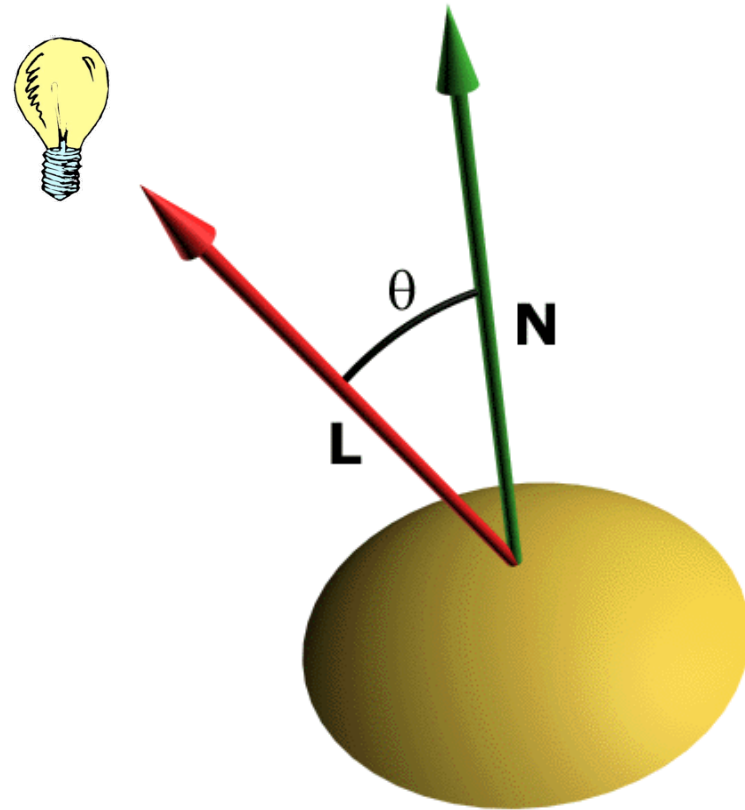
Modellazione della riflessione diffusiva

- ❖ Si approssima la funzione di riflessione diffusa della superficie come una costante k_d dipendente dal materiale
- ❖ Equazione di illuminazione (solo diffusiva)

$$I = I_p k_d \cos \theta$$

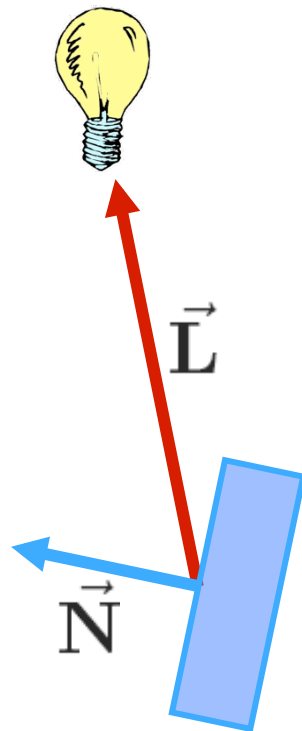
o meglio

$$I = I_p k_d (\vec{\mathbf{N}} \cdot \vec{\mathbf{L}})$$

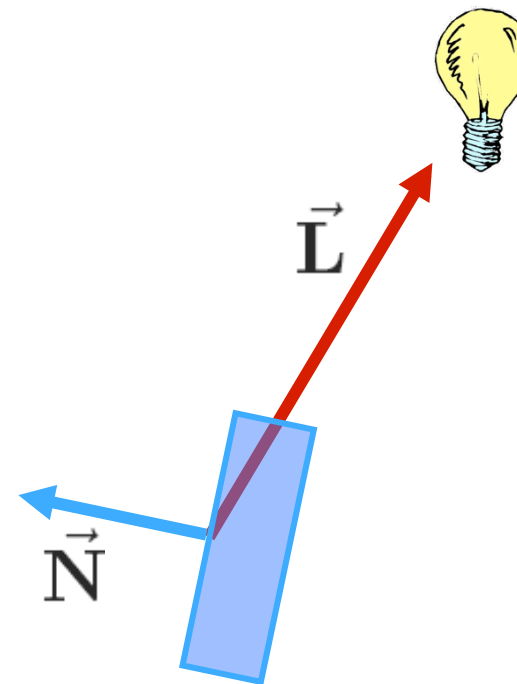


Modellazione della riflessione diffusiva

- ❖ Si considera solo per valori di θ compresi tra 0 e $\pi/2$



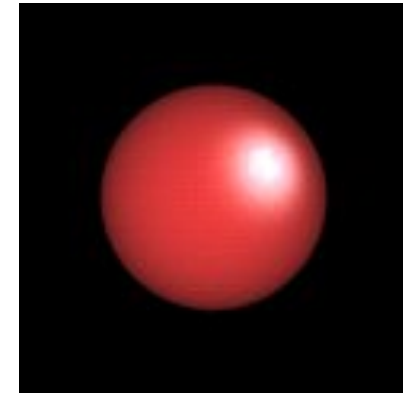
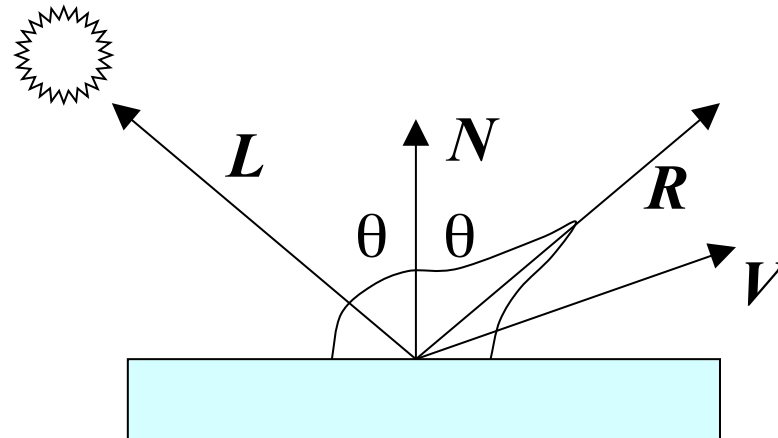
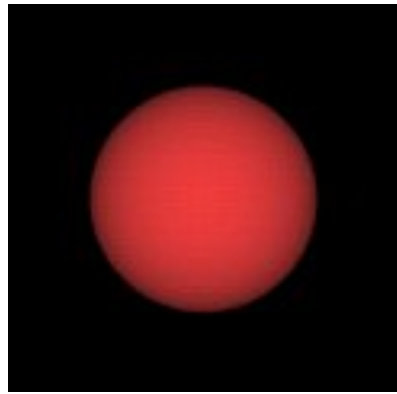
OK



NO

Modellazione della riflessione speculare

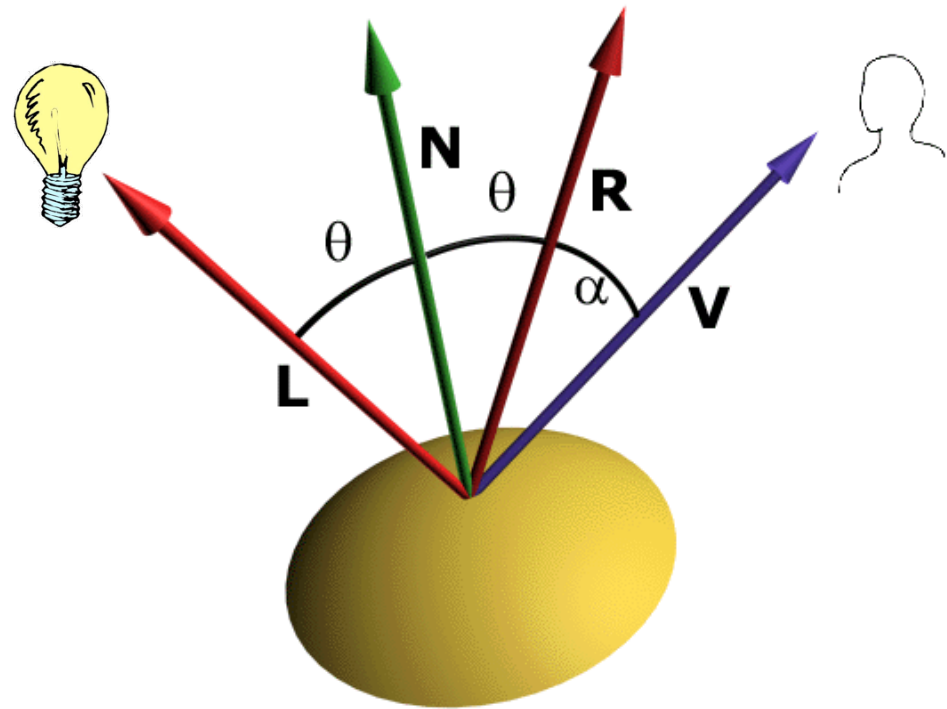
- ❖ Novità sostanziale: *riflettore non perfetto*



- ❖ Approssimazione empirica di una riflessione più realistica rispetto alla legge di Fresnel
- ❖ Conseguenza: *specular highlight*

Modellazione della riflessione speculare

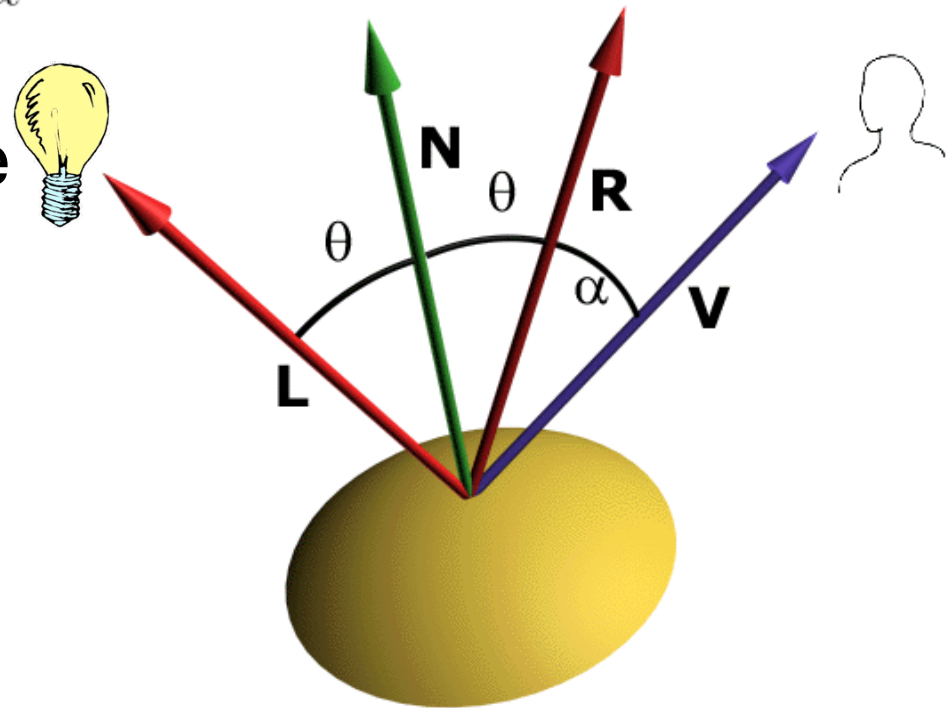
- ❖ Dipendenza dall'angolo α compreso tra la direzione di riflessione ideale e la direzione di vista
- ❖ Riflessione massima per $\alpha = 0$
- ❖ Decadimento più o meno rapido all'aumentare di α



Modellazione della riflessione speculare

- ❖ Questo comportamento si modella elevando alla n il coseno dell'angolo α
- ❖ Il parametro n è detto esponente di riflessione speculare (*specular reflection exponent*) del materiale
- ❖ Il vettore \vec{R} si calcola

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

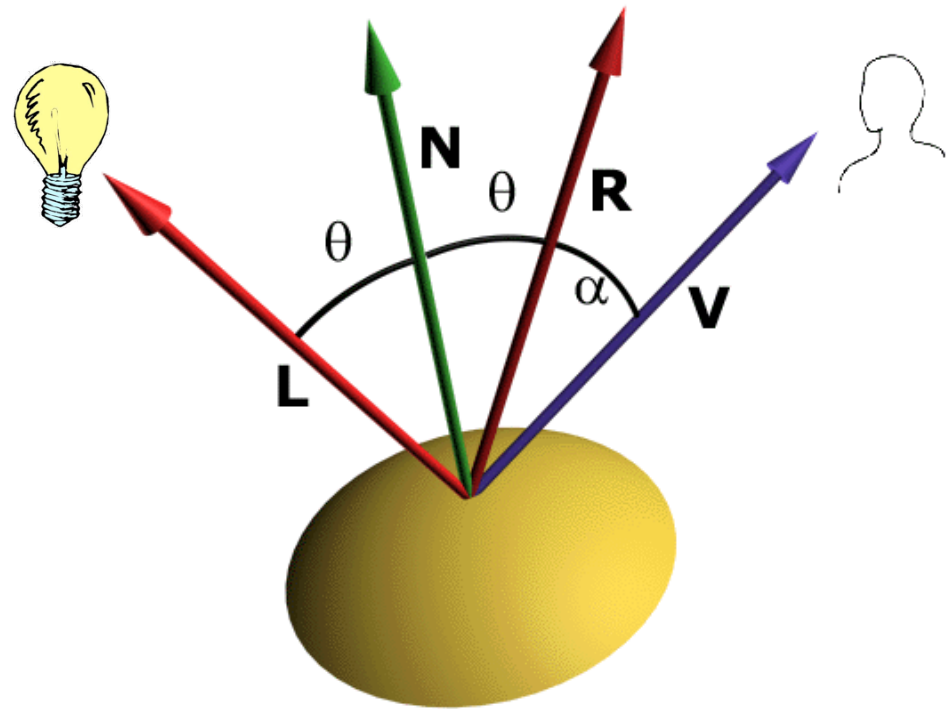


Modellazione della riflessione speculare

- ❖ Equazione di illuminazione (solo speculare)

$$I = I_p k_s \cos^n \alpha$$

- ❖ Parametro k_s modella il comportamento della superficie insieme a n



Modellazione della componente ambientale

- ❖ Le inter-riflessioni tra oggetti diversi nella scena non sono modellate in modo accurato dal modello di Phong

- ❖ Sono approssimate dalla componente:

$$I = I_a k_a$$

- ❖ I_a modella la radiazione luminosa totale emessa nella scena

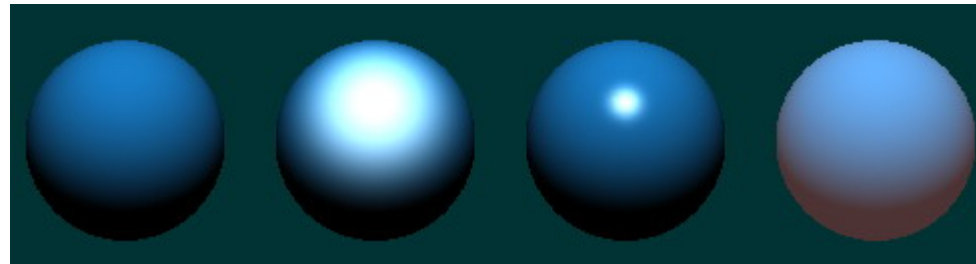
- ❖ k_a modella la riflettività del materiale

- ❖ I_a è costante per tutti i punti di tutti gli oggetti

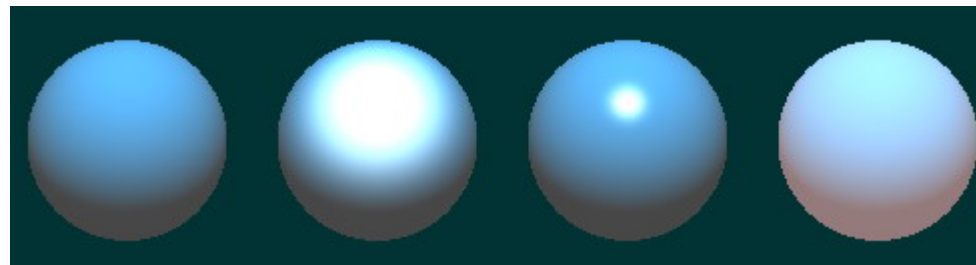
Modellazione della componente ambientale

- ❖ La componente ambientale aggiunge realismo alla scena

Senza

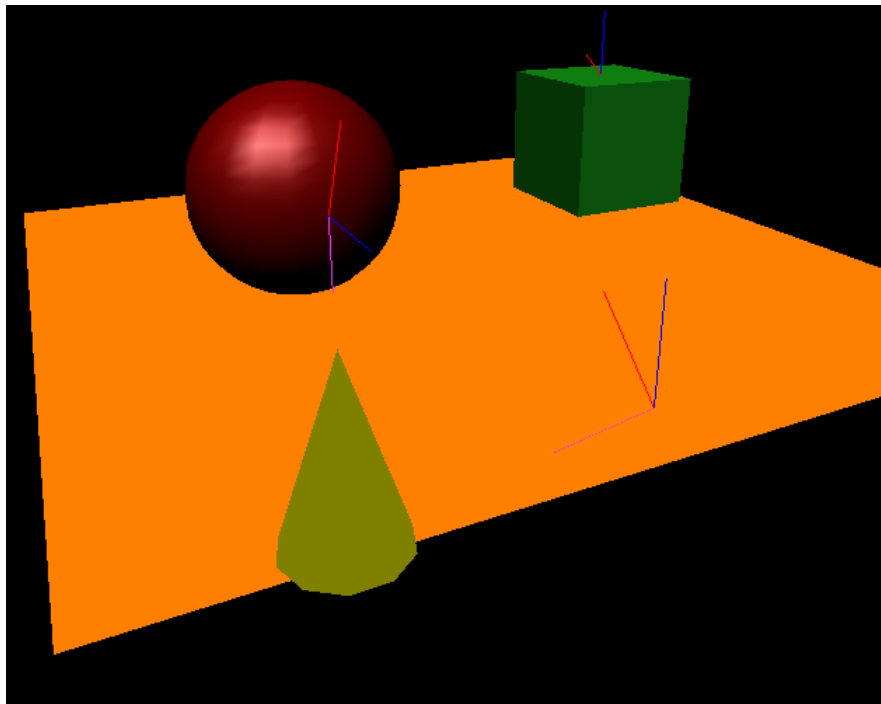


Con

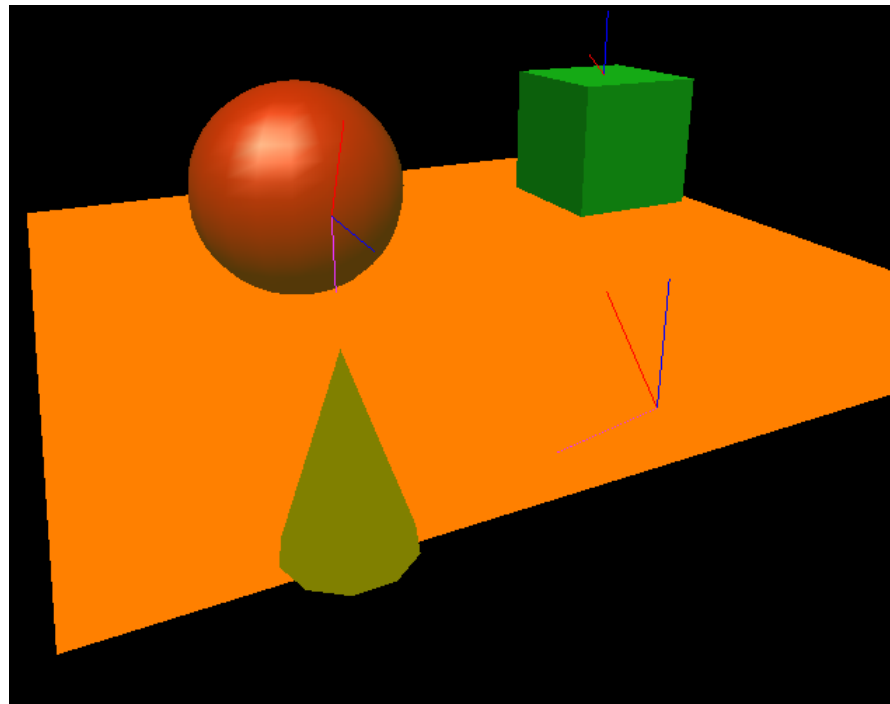


Modellazione della componente ambientale

- ❖ La componente ambientale aggiunge realismo alla scena



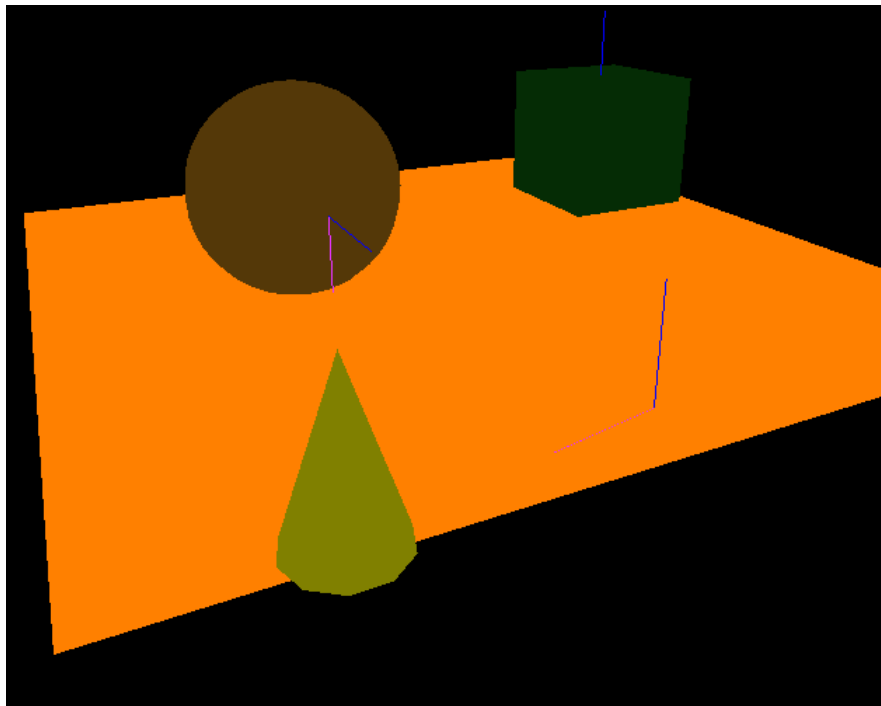
Senza



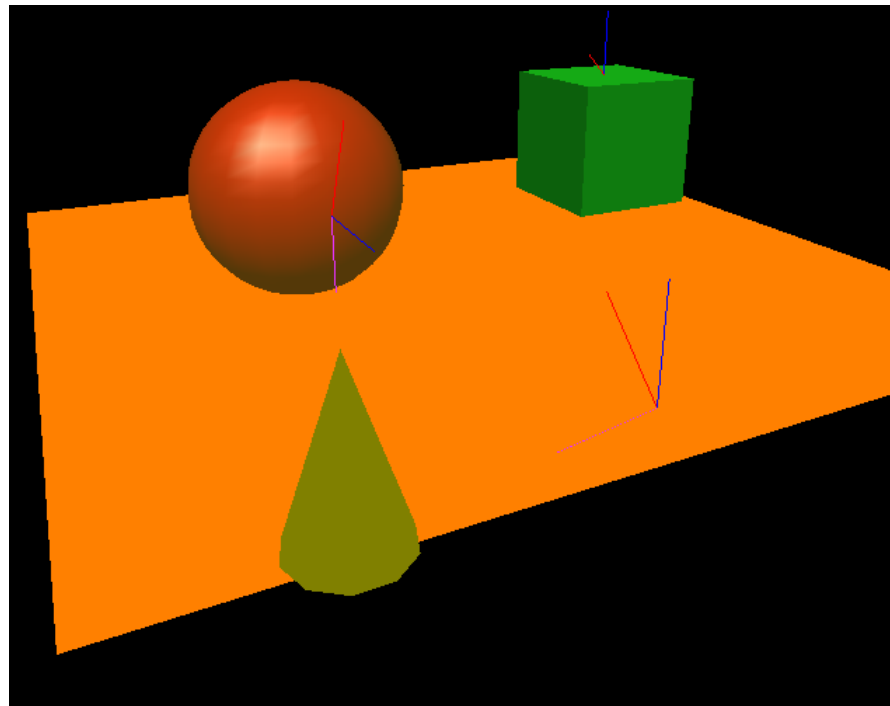
Con

Modellazione della componente ambientale

❖ Ma da sola non basta!



Solo ambientale



Con riflessioni

Mettere tutto insieme

- ❖ Tutti i contributi descritti si vanno a sommare per calcolare l'equazione di illuminazione
- ❖ Sommatoria su tutte le sorgenti luminose presenti nella scena

$$I = I_a k_a + \sum_p I_p [k_d (\vec{\mathbf{N}} \cdot \vec{\mathbf{L}}) + k_s (\vec{\mathbf{R}} \cdot \vec{\mathbf{V}})^n]$$

Mettere tutto insieme + attenuazione

- ❖ Si può tenere conto dell'attenuazione dell'intensità dell'illuminazione all'aumentare della distanza

$$f_{\text{att}} = \begin{cases} \frac{1}{c_1 + c_2 d_L + c_3 d_L^2} & \text{se } \frac{1}{c_1 + c_2 d_L + c_3 d_L^2} < 1 \\ 1 & \text{altrimenti} \end{cases}$$

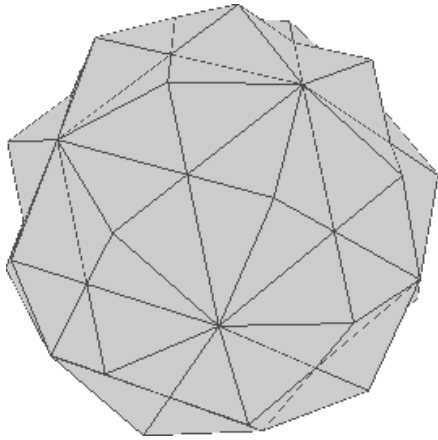
- ❖ Inserendo il fattore di attenuazione

$$I = I_a k_a + \sum_p f_{\text{att}_p} I_p [k_d (\vec{\mathbf{N}} \cdot \vec{\mathbf{L}}) + k_s (\vec{\mathbf{R}} \cdot \vec{\mathbf{V}})^n]$$

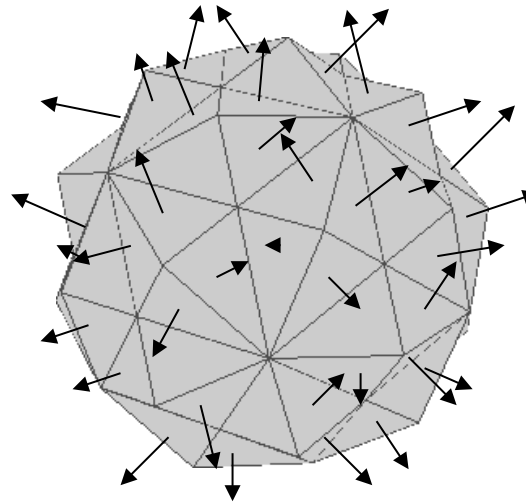
Tecniche di shading

- ❖ Il modello di Phong descrive *come* deve essere calcolata l'interazione tra luce e materia
- ❖ Dobbiamo capire *dove* calcolare l'equazione d'illuminazione
- ❖ Sistema interattivo \Rightarrow generazione di un certo numero di frame per secondo \Rightarrow metodi approssimati

Shading costante

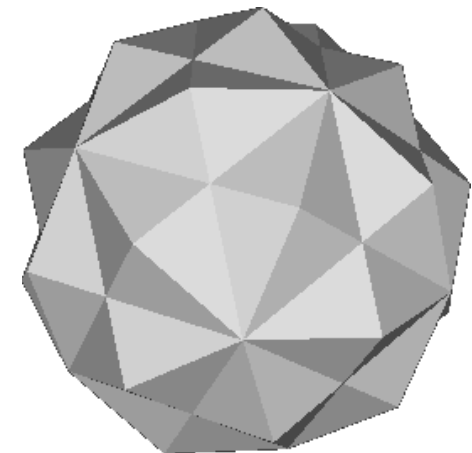


Dato l'oggetto per cui calcolare l'equazione di illuminazione I ...



...calcolare le normali in ogni faccia...

...e calcolo I una sola volta per faccia



Shading costante

❖ Se:

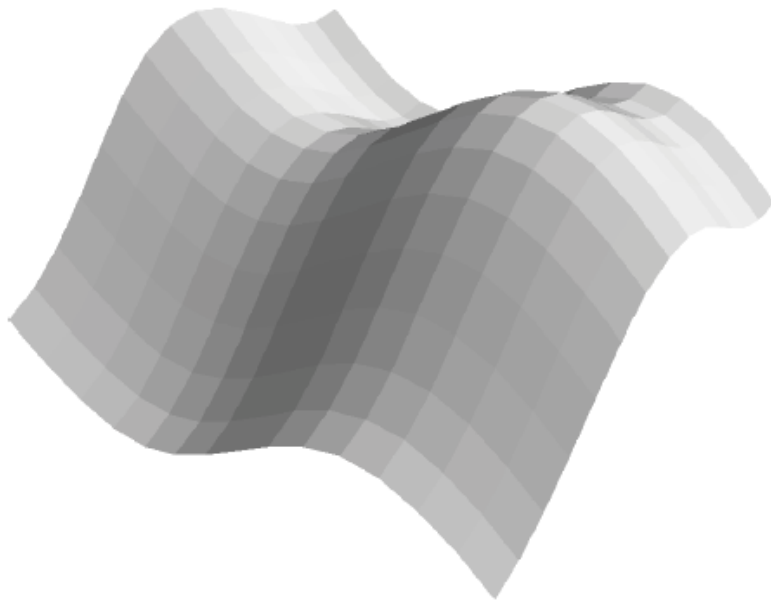
❖ sorgenti luminose solo direzionali ($|\vec{N} \cdot \vec{L}| = k$ per tutta la superficie)

❖ osservatore a distanza infinita dalla scena (proiezione parallela) $\Rightarrow \vec{N} \cdot \vec{V} = k$ e $\vec{R} \cdot \vec{V} = k$ per tutta la superficie

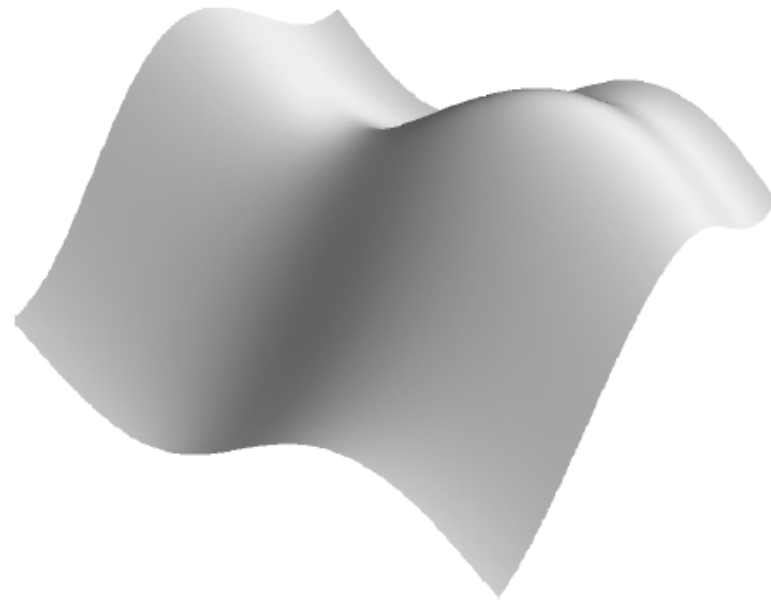
il metodo è la migliore approssimazione possibile

Shading costante

- ❖ Problema: il modello discreto rappresenta in modo approssimato una superficie curva e continua



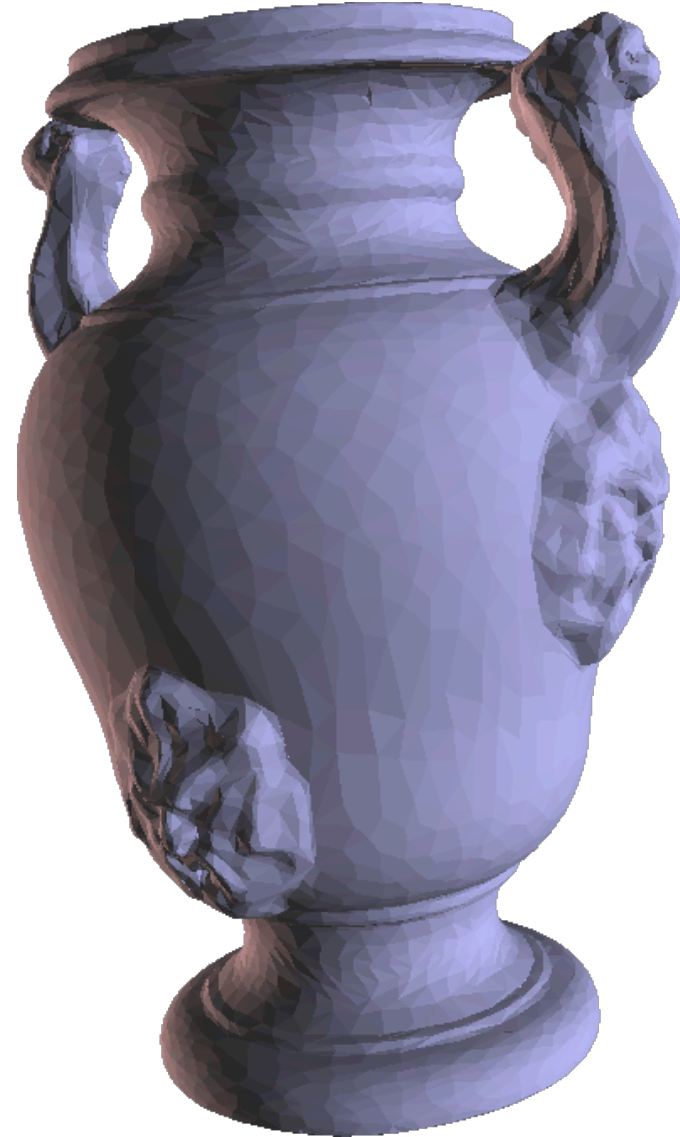
Com'è



Come dovrebbe essere

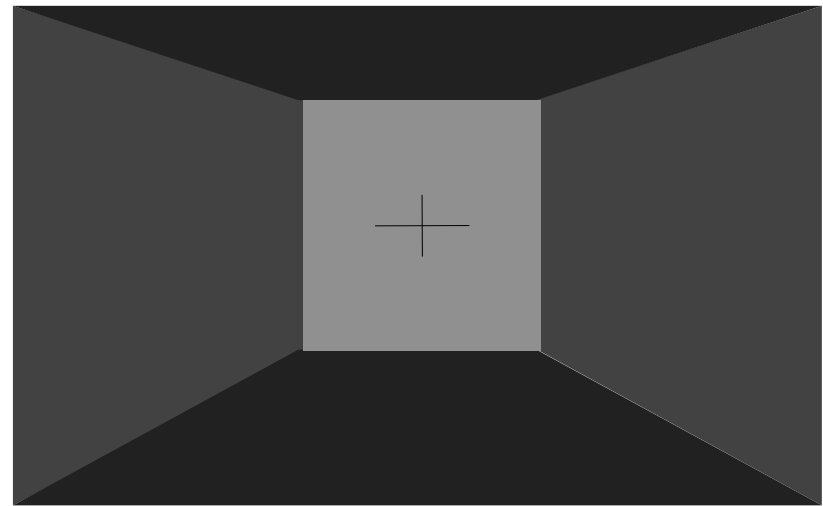
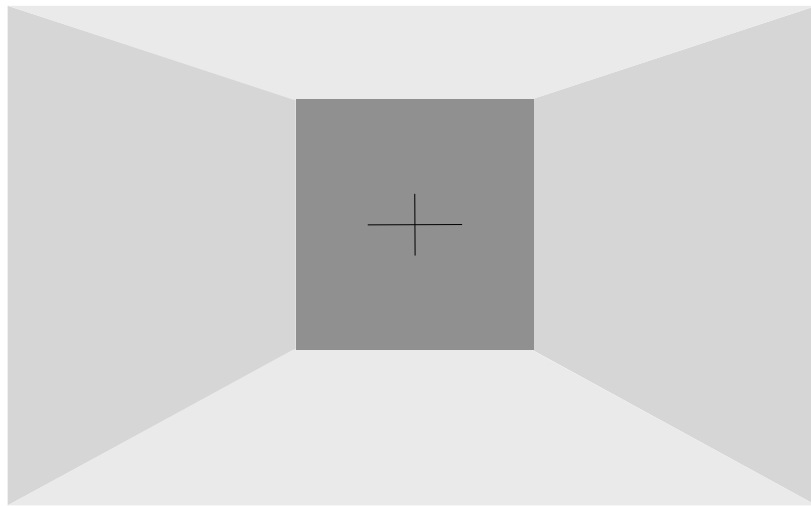
Shading costante

- ❖ Soluzione: uso un numero elevato di facce
- ❖ Non funziona, si vedono comunque le discontinuità tra una faccia e la vicina



Mach banding

- ❖ Alterazione della percezione visiva di una zona in cui la luminanza varia rapidamente
- ❖ Un oggetto messo vicino ad uno più chiaro risulta più scuro e messo vicino ad uno più scuro risulta più chiaro

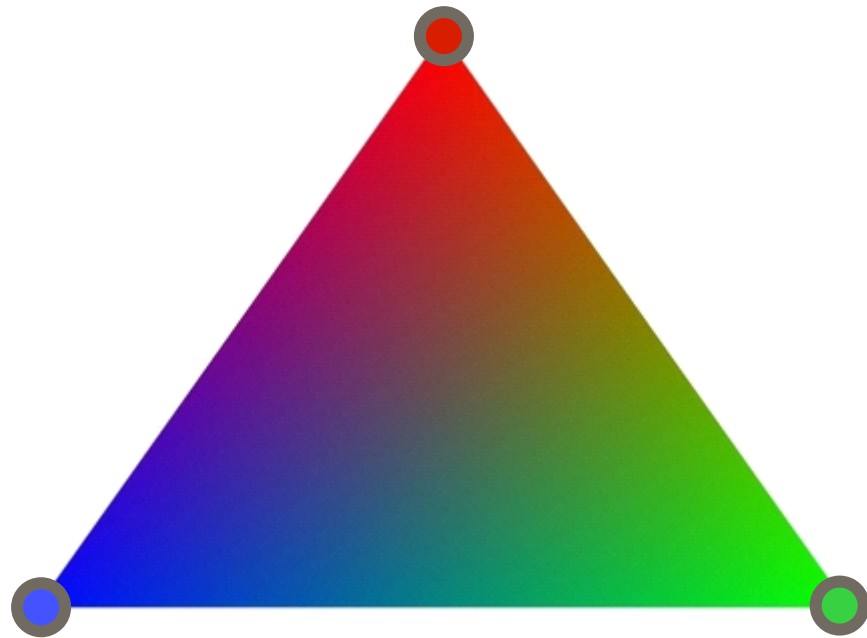


Gouraud shading

- ❖ Proprietà fondamentale dello spazio colore RGB: linearità
- ❖ Il valore colore intermedio tra due colori dati nello spazio RGB si calcola per interpolazione lineare
- ❖ Interpolazione separata sulle tre componenti R, G, e B

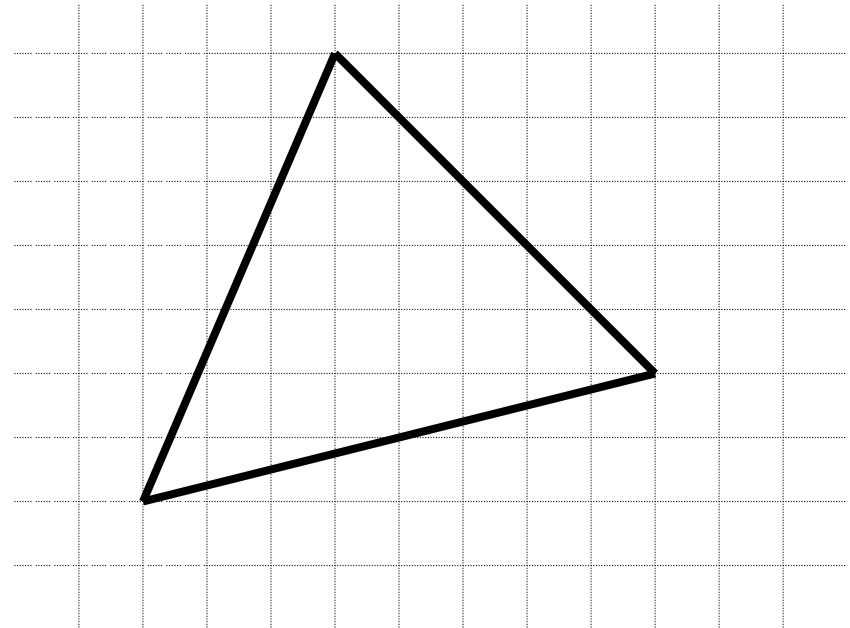
Gouraud shading

- ❖ Calcolare l'equazione di illuminazione solo in alcuni punti nodali
- ❖ Interpolare linearmente tra questi valori



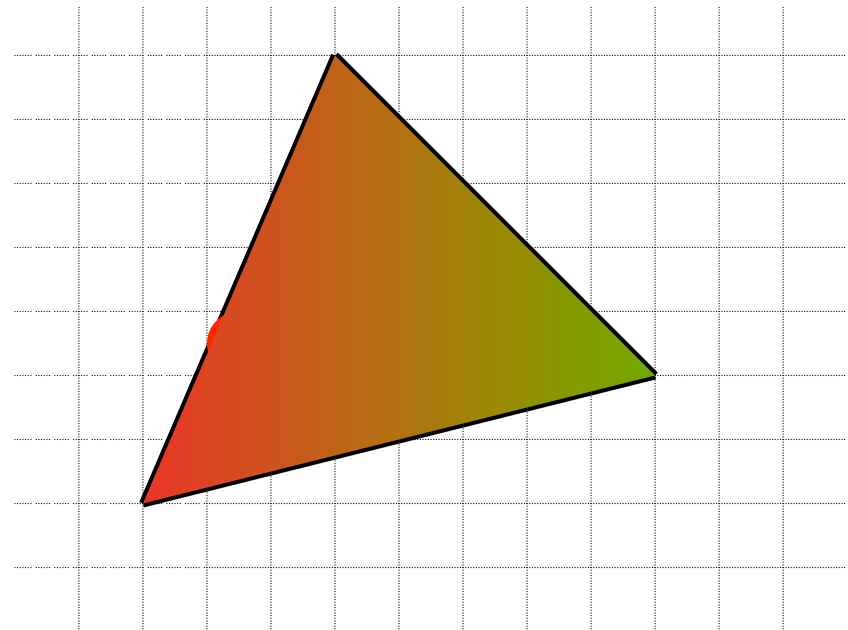
Gouraud shading

- ❖ Aggiungere all'algoritmo di rasterizzazione l'operazione di interpolazione nello spazio colore comporta uno sforzo minimo



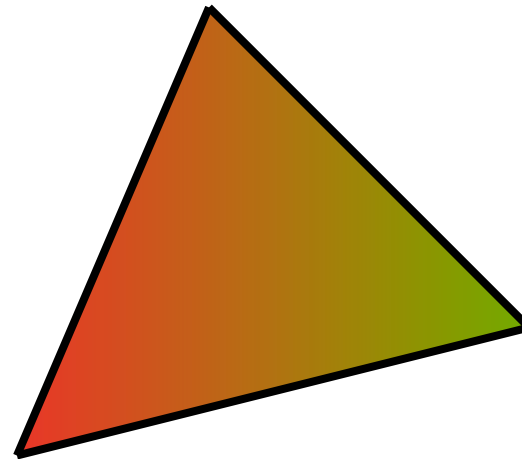
Gouraud shading

- ❖ Per ogni span si calcola il valore di I all'estremo con un algoritmo incrementale, e, sempre incrementalmente, si calcolano i valori all'interno della span



Gouraud shading

- ❖ Il risultato così ottenuto approssima molto il modello di Phong per superfici generiche rispetto allo shading costante

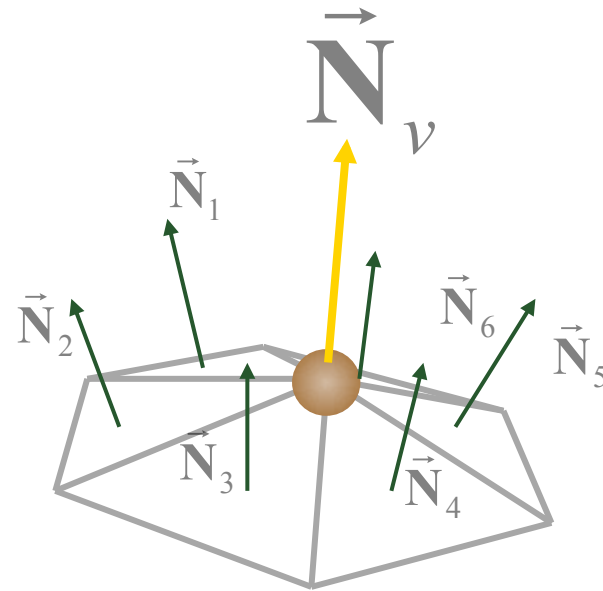


Gouraud shading

- ❖ Che normali utilizzo?
- ❖ La normale alla faccia è bene definita
- ❖ La normale al vertice la calcolo come media delle normali delle facce che

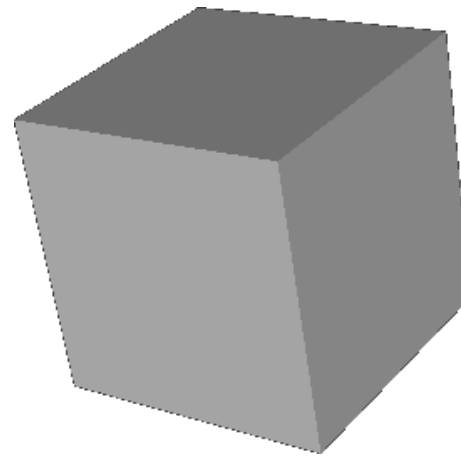
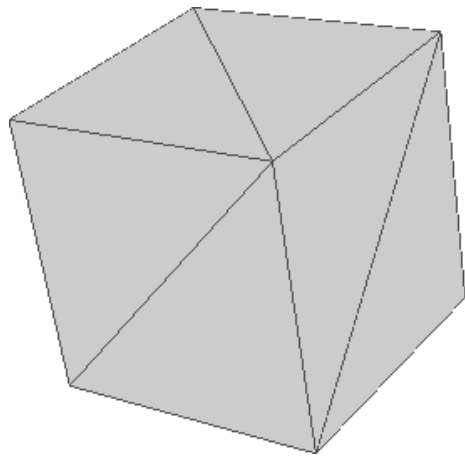
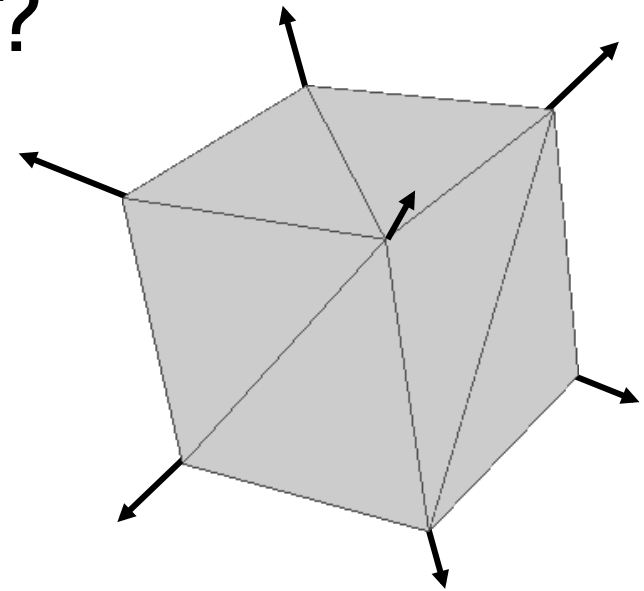
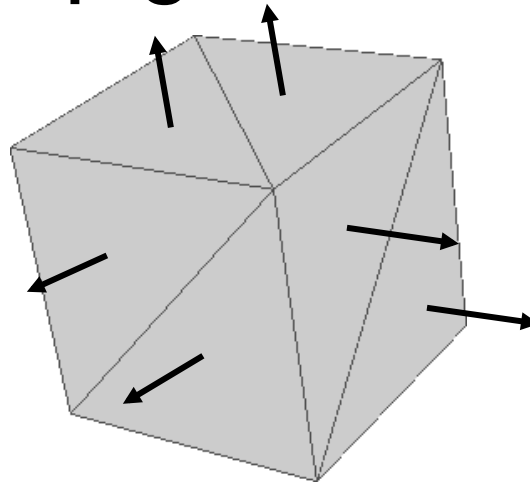
insie

$$\vec{N}_v = \frac{\sum_i \vec{N}_i}{|\sum_i \vec{N}_i|} \quad \text{ce}$$

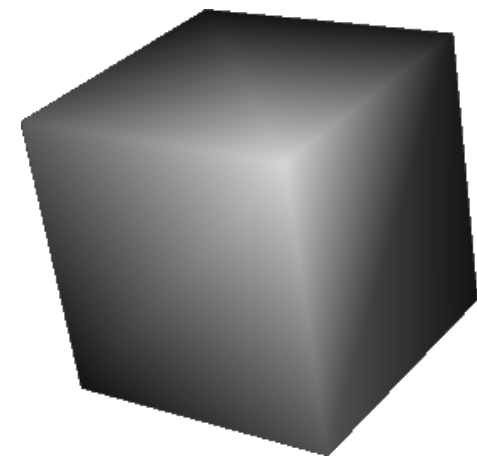


Gouraud shading

❖ Problema: gli spigoli “veri”?



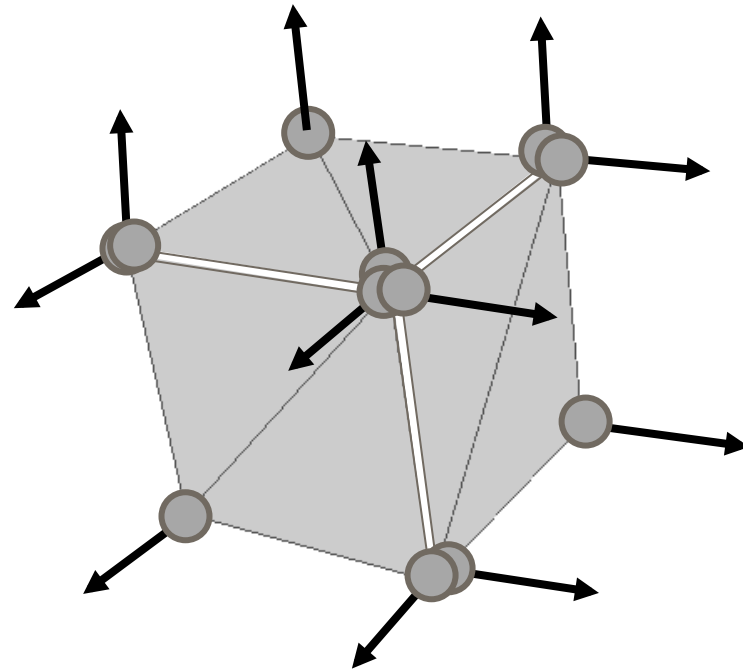
shading costante



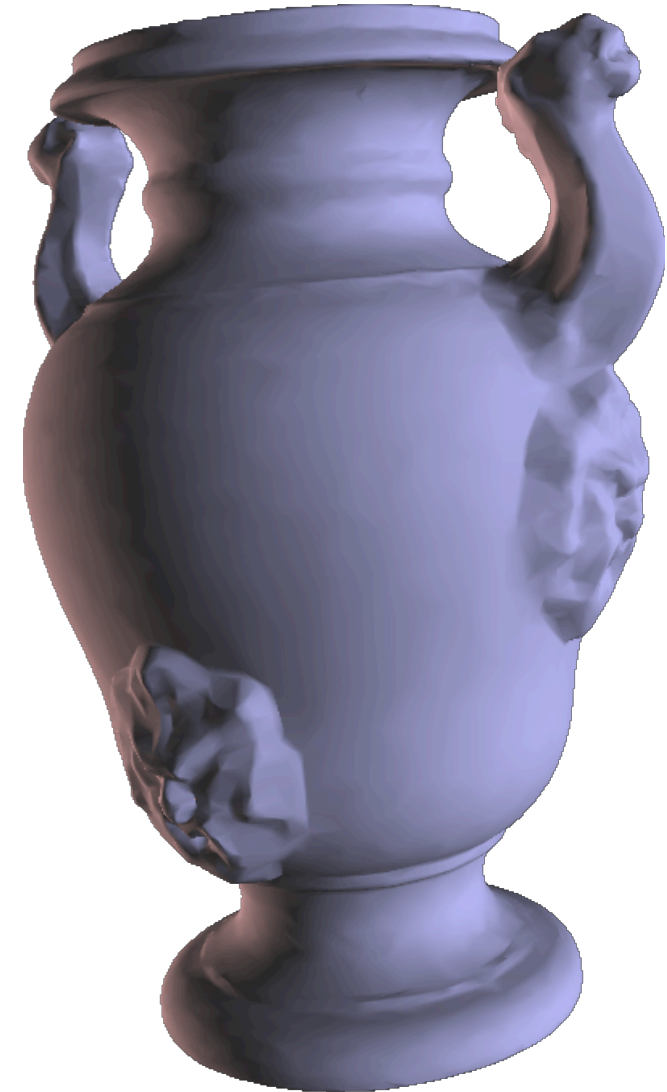
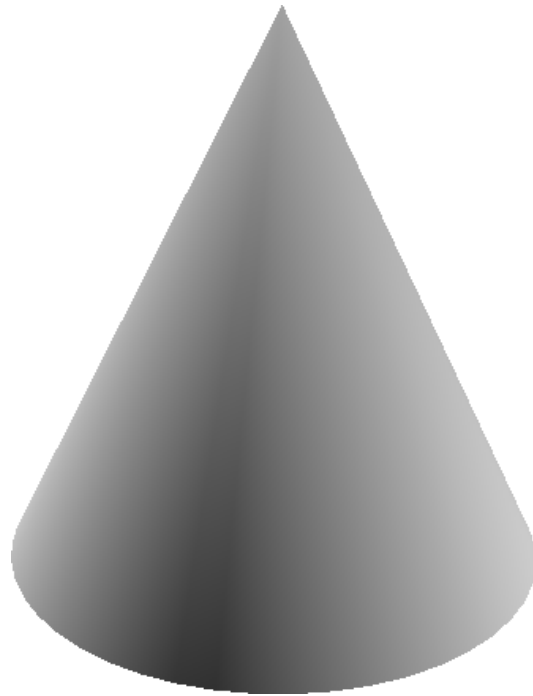
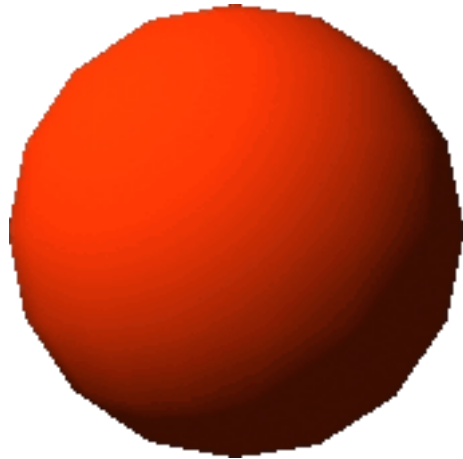
Gouraud shading

Gouraud shading

- ❖ Soluzione: si utilizzano normali diverse per i due lati dello spigolo
- ❖ La struttura dati deve memorizzare le adiacenze e le diverse tipologie



Paragone: costante e Gouraud

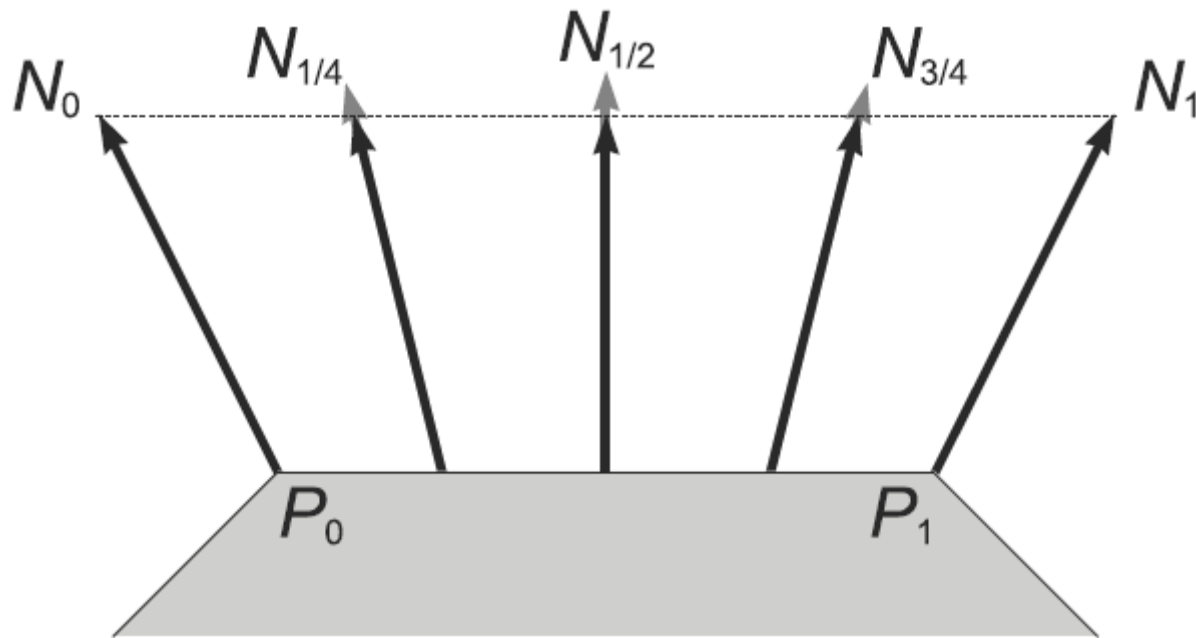


Phong shading

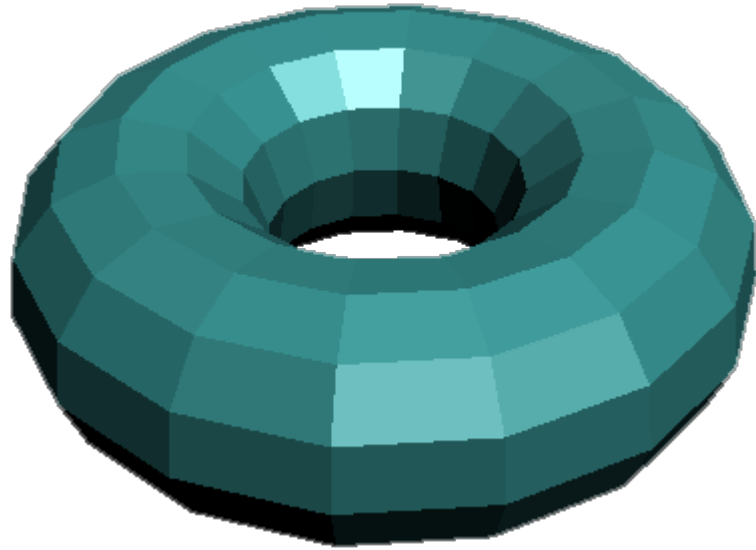
- ❖ Gouraud shading: ottimale rapporto qualità/prezzo
- ❖ Risultati non eccezionali per superfici dotate di un alto coefficiente di riflessione speculare
- ❖ Problema: per n alto lo *specular highlight* deve essere piccolo, invece si “propaga” per tutta la faccia (per interpolazione) se cade vicino a un vertice, si “perde” se è interno

Phong shading

- ❖ Soluzione: si interpola nello spazio delle normali e si calcola l'equazione di illuminazione in ogni pixel



Paragone: costante, Gouraud e Phong



Costante



Phong



Gouraud

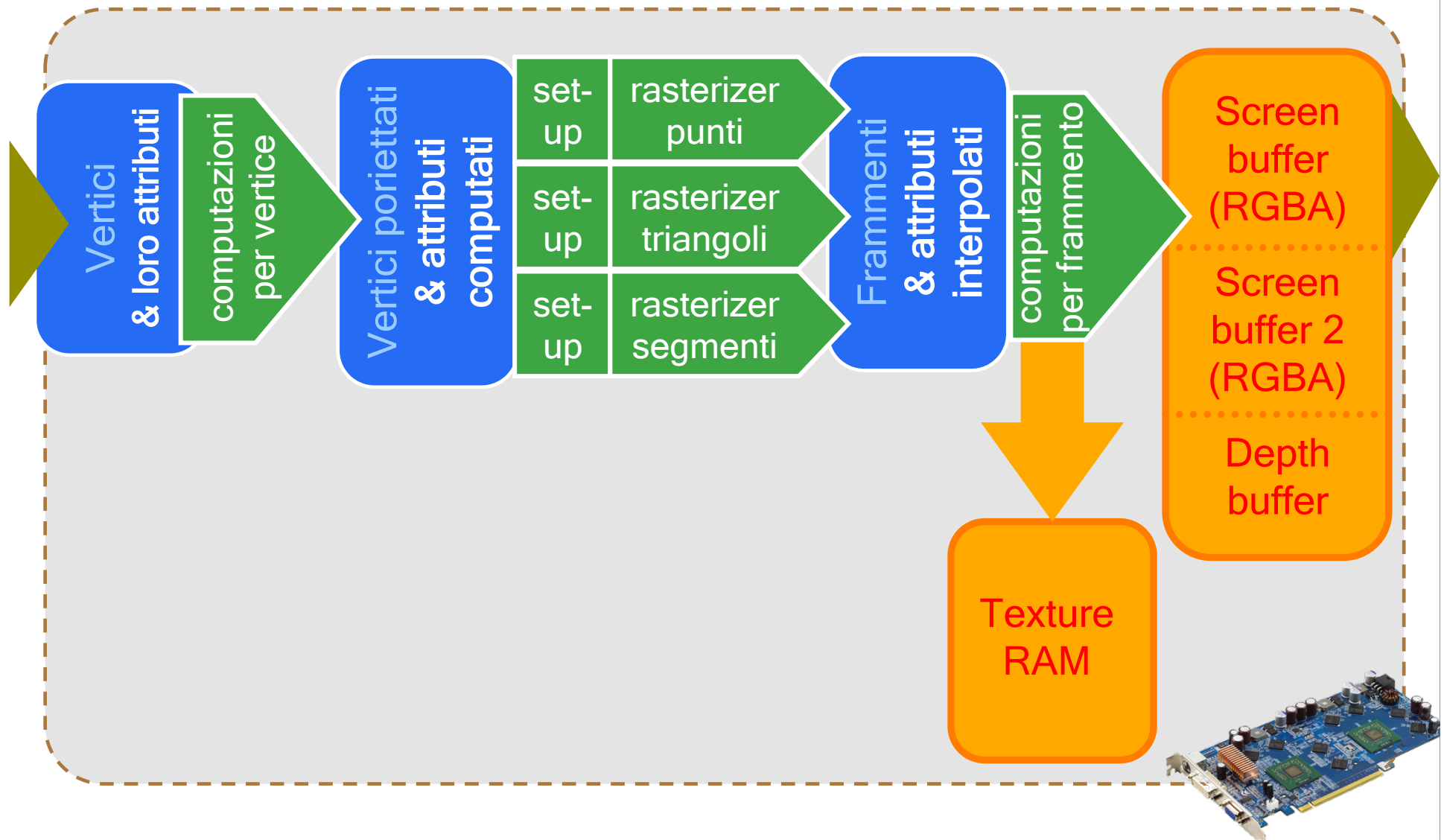
Texture mapping

- ❖ Shading: funziona finché modello geometrico ha complessità della composizione del materiale
- ❖ La descrizione di un materiale non uniforme è corretta a partire da una rappresentazione geometrica con suddivisione (*tassellazione*) in primitive collegata alle discontinuità del materiale da rappresentare
- ❖ Altrimenti? Texture mapping!

Texture mapping

- ❖ Shading: funziona finché modello geometrico ha complessità della composizione del materiale
- ❖ La descrizione di un materiale non uniforme è corretta a partire da una rappresentazione geometrica con suddivisione (**tassellazione**) in primitive collegata alle discontinuità del materiale da rappresentare
- ❖ Altrimenti? Texture mapping!

Memoria RAM nelle schede grafiche



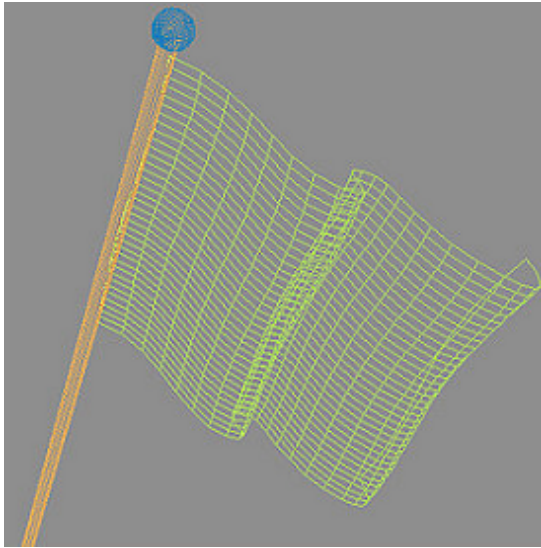
Texture Mapping

- ❖ Nelle operazioni per frammento si può accedere ad una RAM apposita: la **Texture RAM** strutturata in un insieme di Textures (“**tessiture**”)
- ❖ Ogni tessitura è un array 1D, 2D o 3D di Texels (campioni di tessitura) dello stesso tipo

Texels

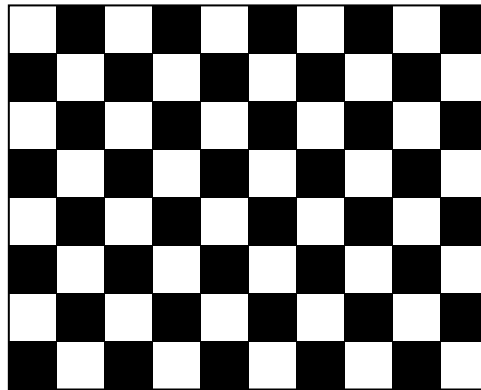
- ❖ Sono esempi di texels:
 - ❖ Ogni texel un colore (componenti: R-G-B, o R-G-B-A): la tessitura è una “color-map”
 - ❖ Ogni texel una componente alpha: la tessitura è una “alpha-map”
 - ❖ Ogni texel una normale (componenti: X-Y-Z): la tessitura è una “normal-map” o “bump-map”
 - ❖ Ogni texel contiene un valore di specularità: la tessitura è una “shininess-map”

Rimappare immagini sulla geometria



geometria 3D
(mesh di quadrilateri)

+

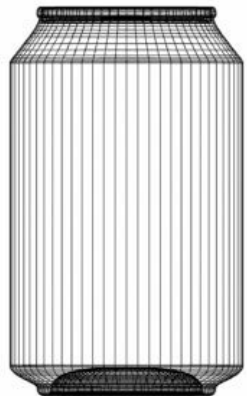
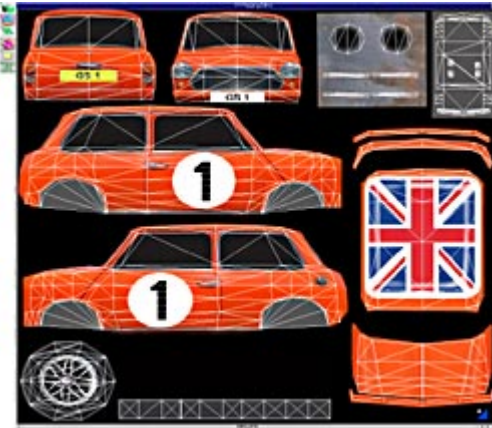


RGB texture 2D
(color-map)

=



Rimappare immagini sulla geometria



Rimappare immagini sulla geometria



+



=



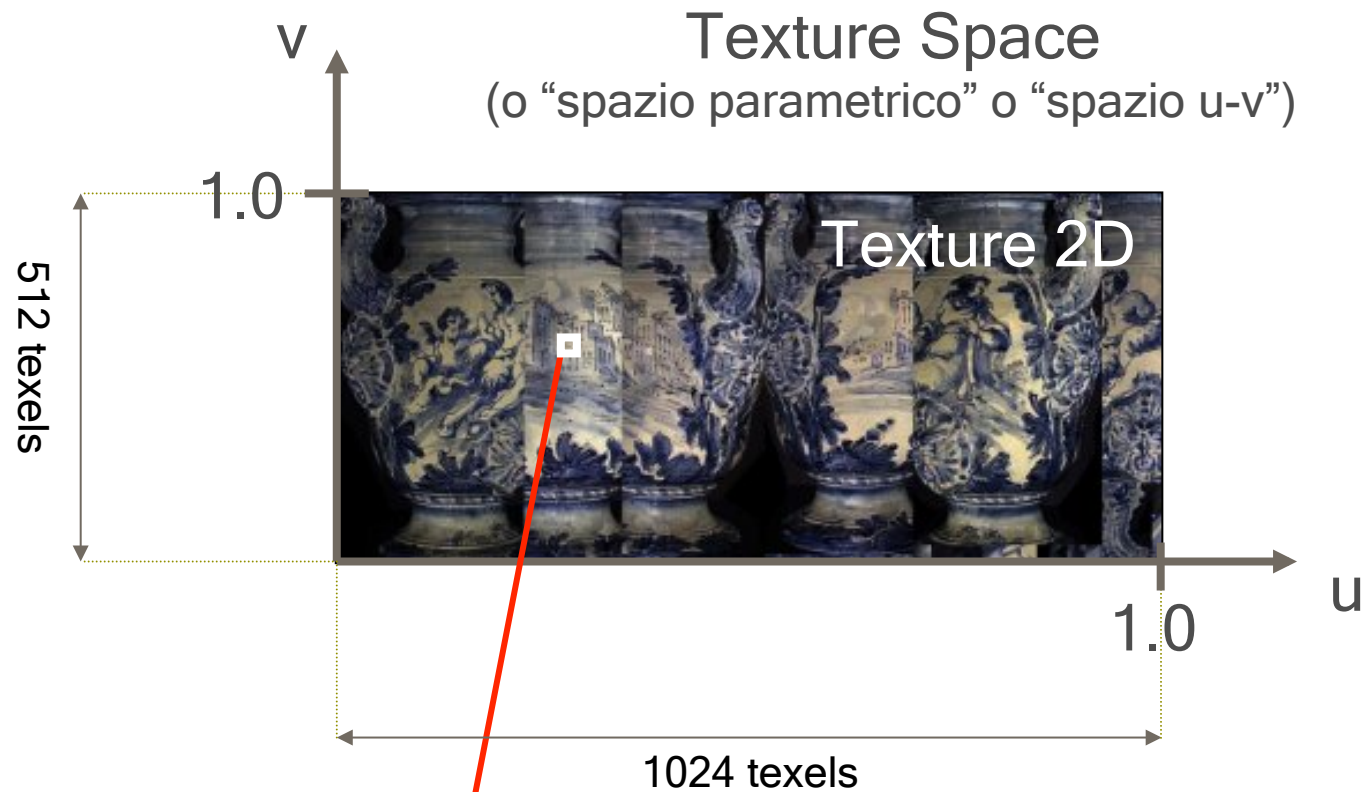
Texture Mapping: storia



Ed Catmull

- ❖ 1974 introdotto da Ed Catmull
 - ❖ nella sua Phd Thesis
- ❖ Solo nel 1992 (!) si ha texture mapping in hardware
 - ❖ Silicon Graphics RealityEngine
- ❖ Dal '92 a oggi ha avuto aumento rapidissimo della diffusione
 - ❖ strada intrapresa soprattutto da low end graphic boards
- ❖ Oggi è una delle più fondamentali tecniche di rendering
 - ❖ Il re indiscusso delle tecniche image based

Notazione

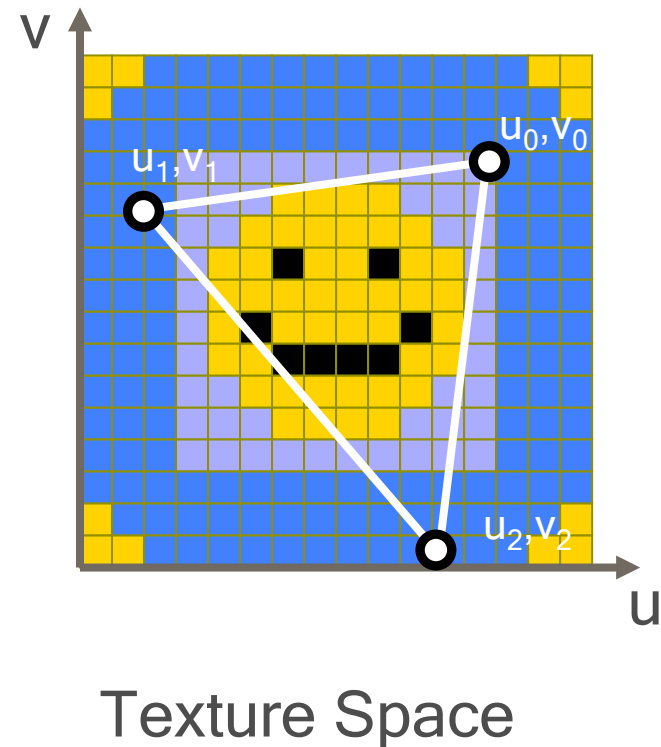
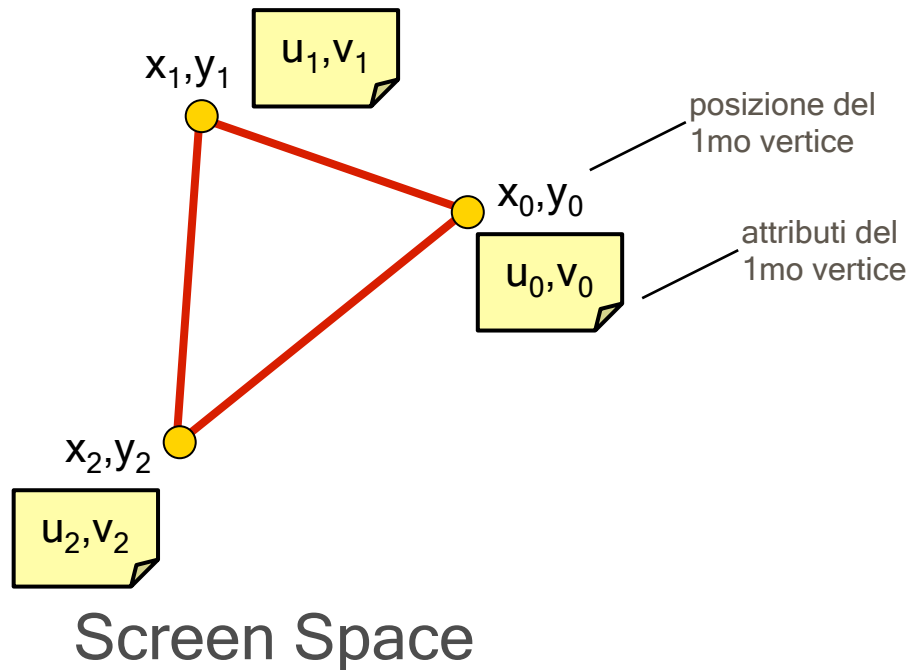


texel

Una Texture è definita nella regione $[0,1] \times [0,1]$ dello "spazio parametrico"

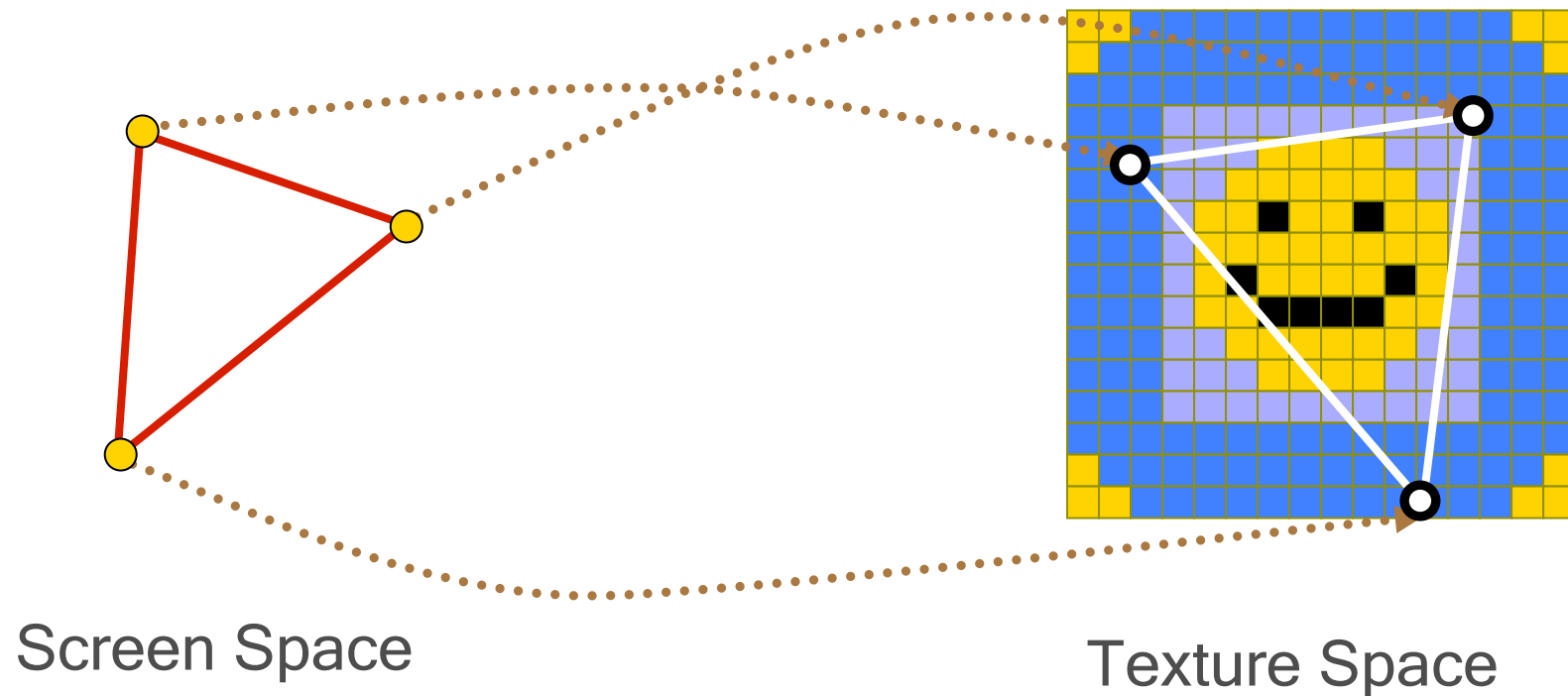
Texture Mapping

- ❖ Ad ogni **vertice** (di ogni triangolo) assegno le sue coordinate u, v nello **spazio tessitura**



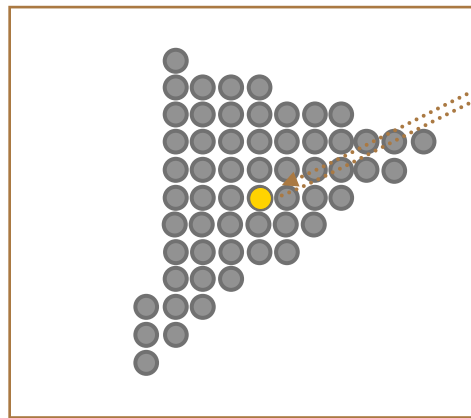
Texture Mapping

- ❖ Così in pratica definisco un **mapping** fra il triangolo e un triangolo di tessitura



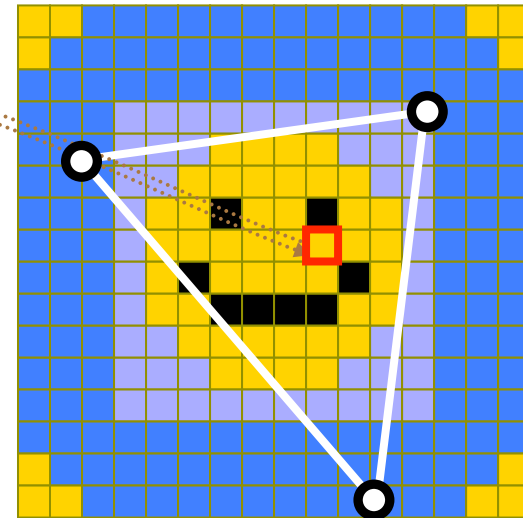
Texture Mapping

- ❖ Ogni **vertice** (di ogni triangolo) ha le sue coordinate u, v nello **spazio tessitura**



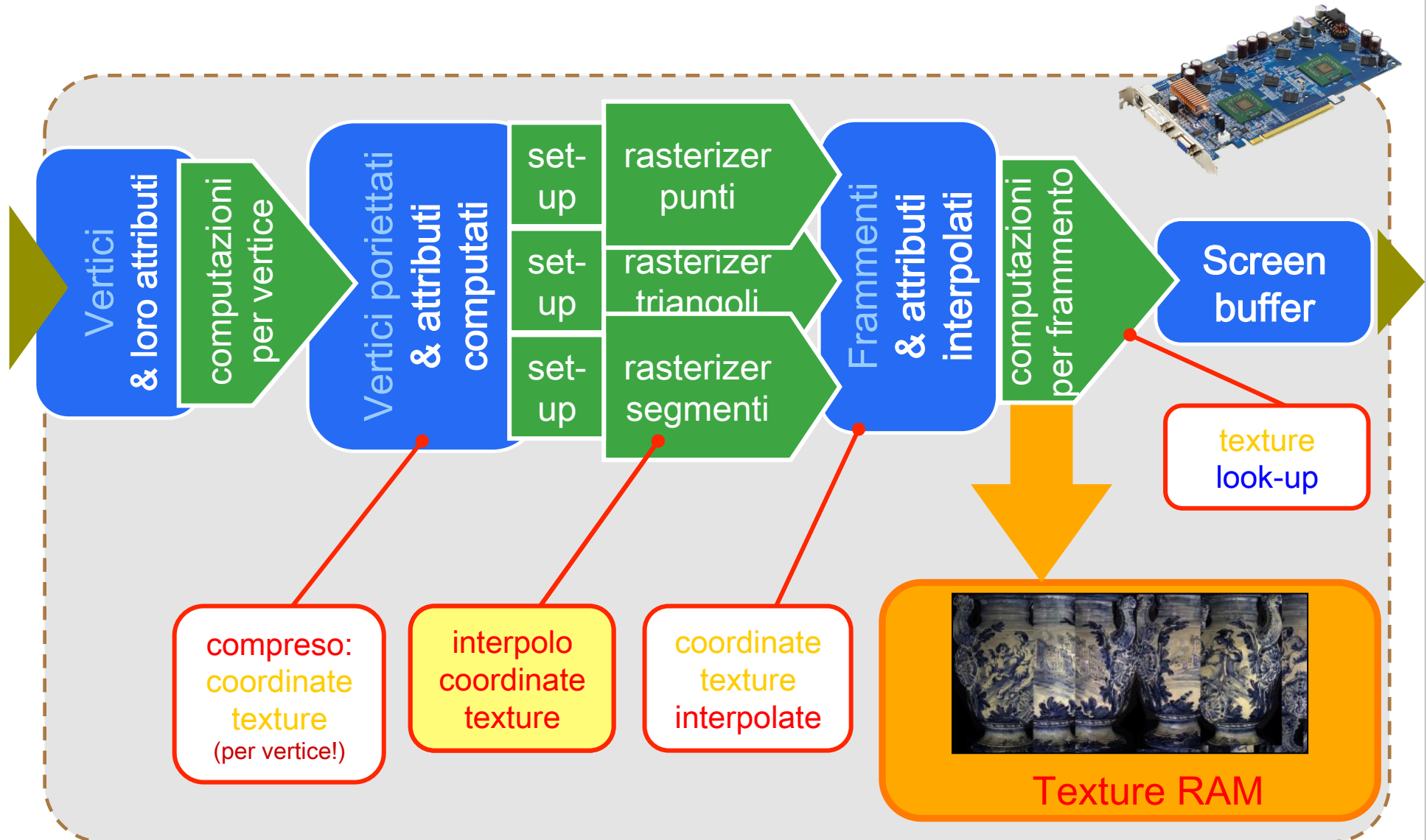
Screen Space

texture look-up

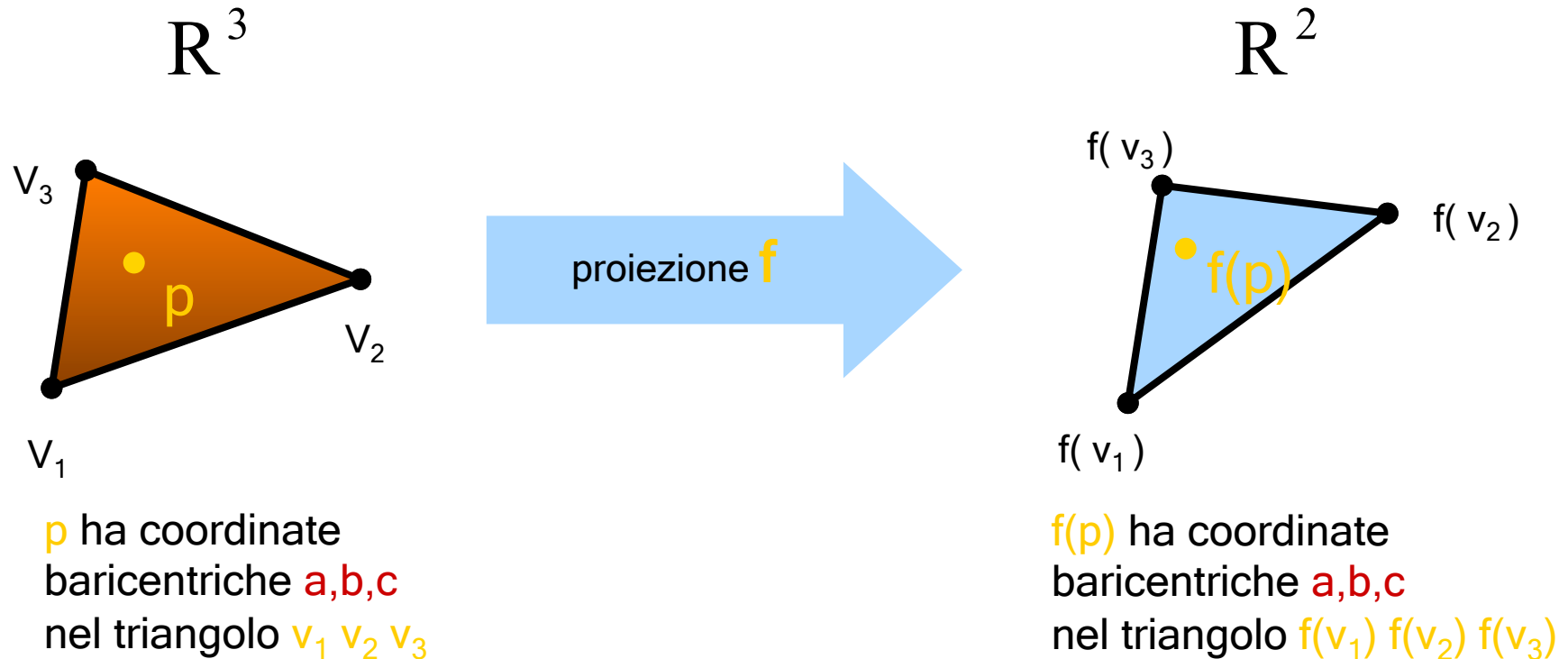


Texture Space

Texture Mapping



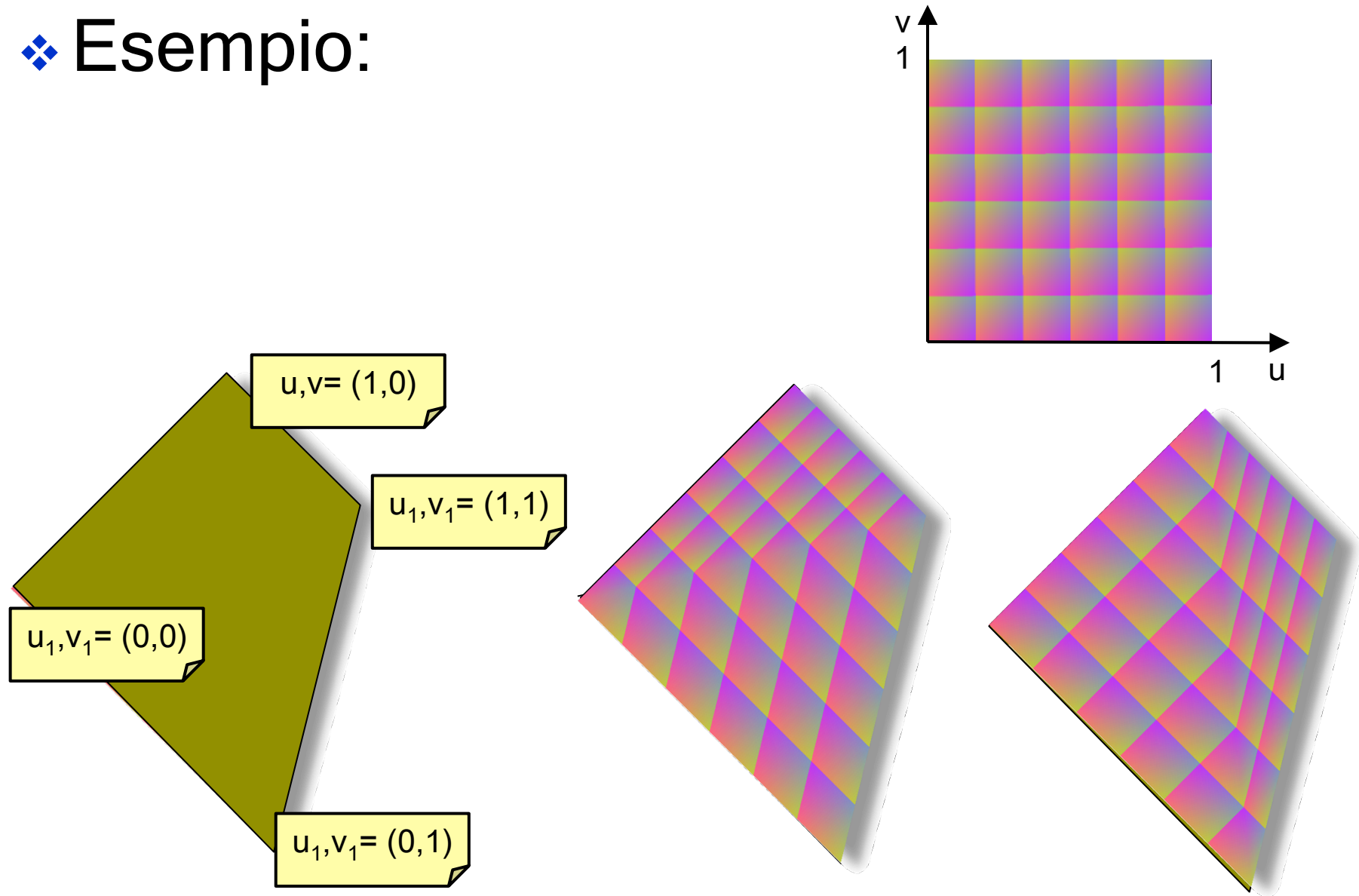
Interpolazione delle coordinate texture



- ❖ Non vale per la proiezione prospettica poiché è solo una approssimazione che è utile per colori e normali ma non funziona quando interpoliamo coordinate texture...

Interpolazione delle coordinate texture

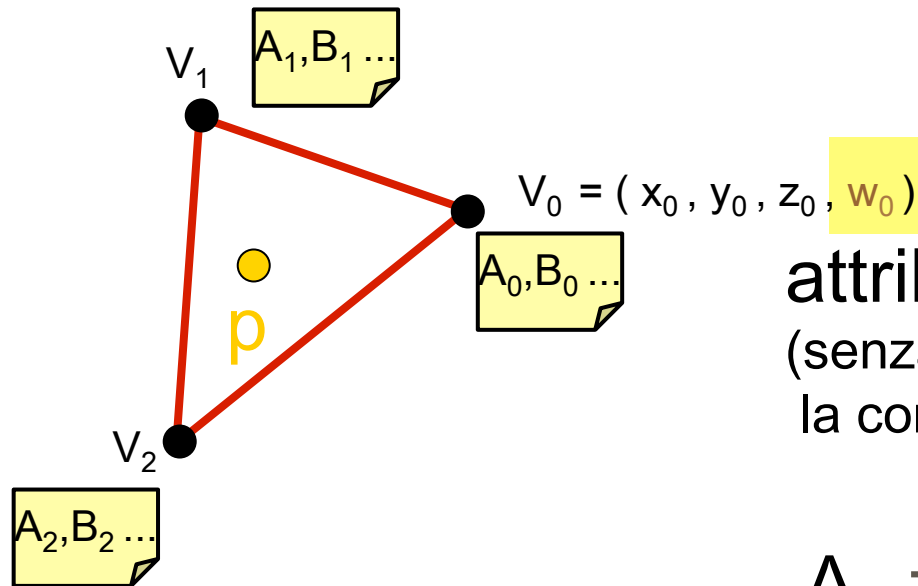
❖ Esempio:



Correzione Prospettica

- ❖ p ha coordinate baricentriche $c_0 c_1 c_2$

$$p = c_0 v_0 + c_1 v_1 + c_2 v_2$$



attributi di p :
(senza considerare
la correzione prospettica)

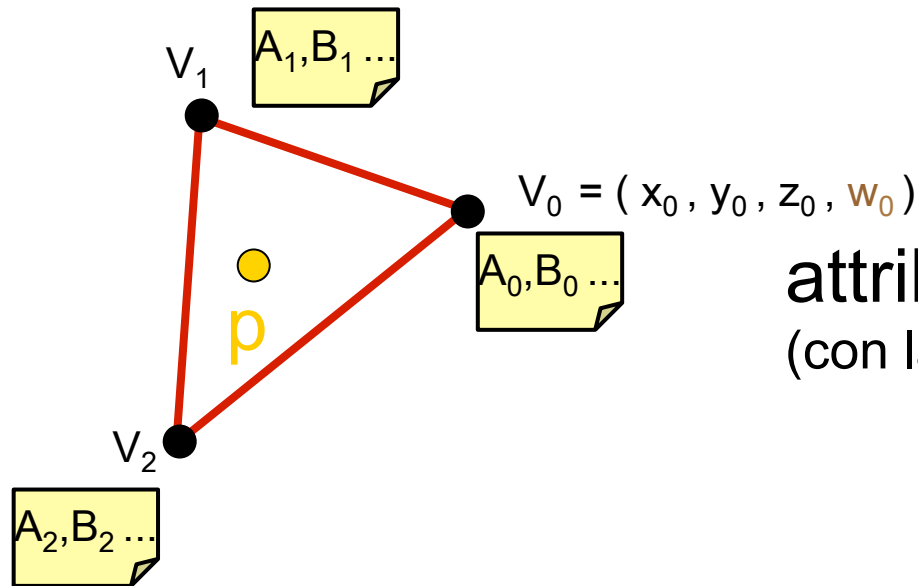
$$A_p = c_0 A_0 + c_1 A_1 + c_2 A_2$$

$$B_p = c_0 B_0 + c_1 B_1 + c_2 B_2$$

Correzione Prospettica

- ❖ p ha coordinate baricentriche $c_0 c_1 c_2$

$$p = c_0 v_0 + c_1 v_1 + c_2 v_2$$

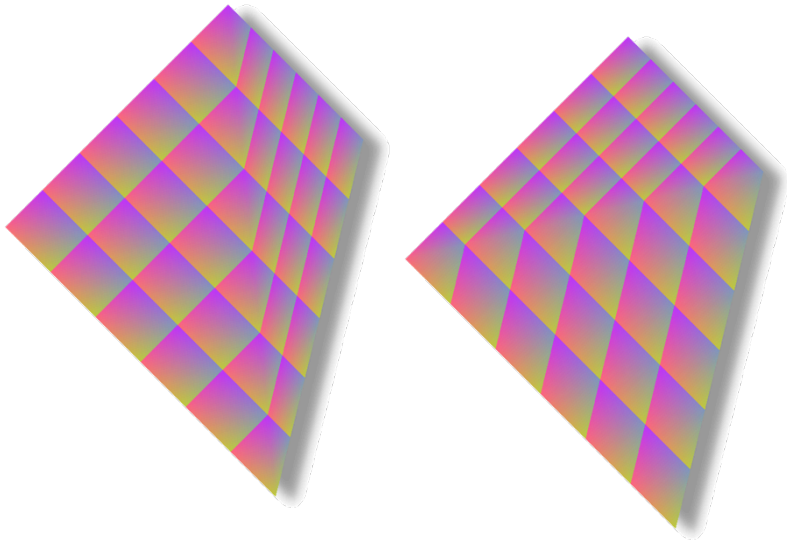


attributi di p :
(con la correzione prospettica)

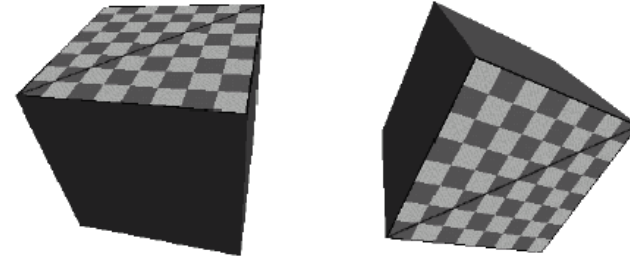
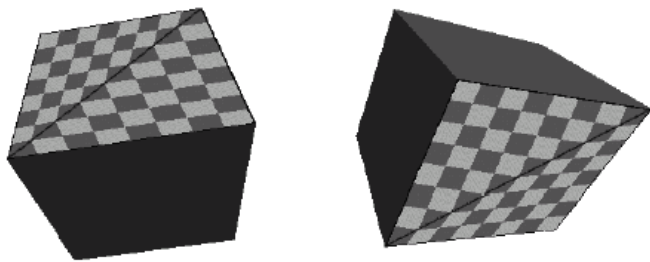
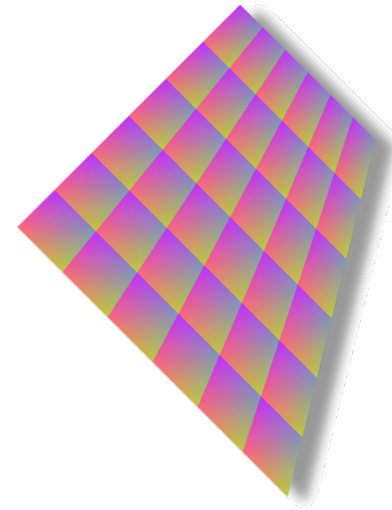
$$A_p = \frac{c_0 \frac{A_0}{w_0} + c_1 \frac{A_1}{w_1} + c_2 \frac{A_2}{w_2}}{c_0 \frac{1}{w_0} + c_1 \frac{1}{w_1} + c_2 \frac{1}{w_2}}$$

Correzione Prospettica

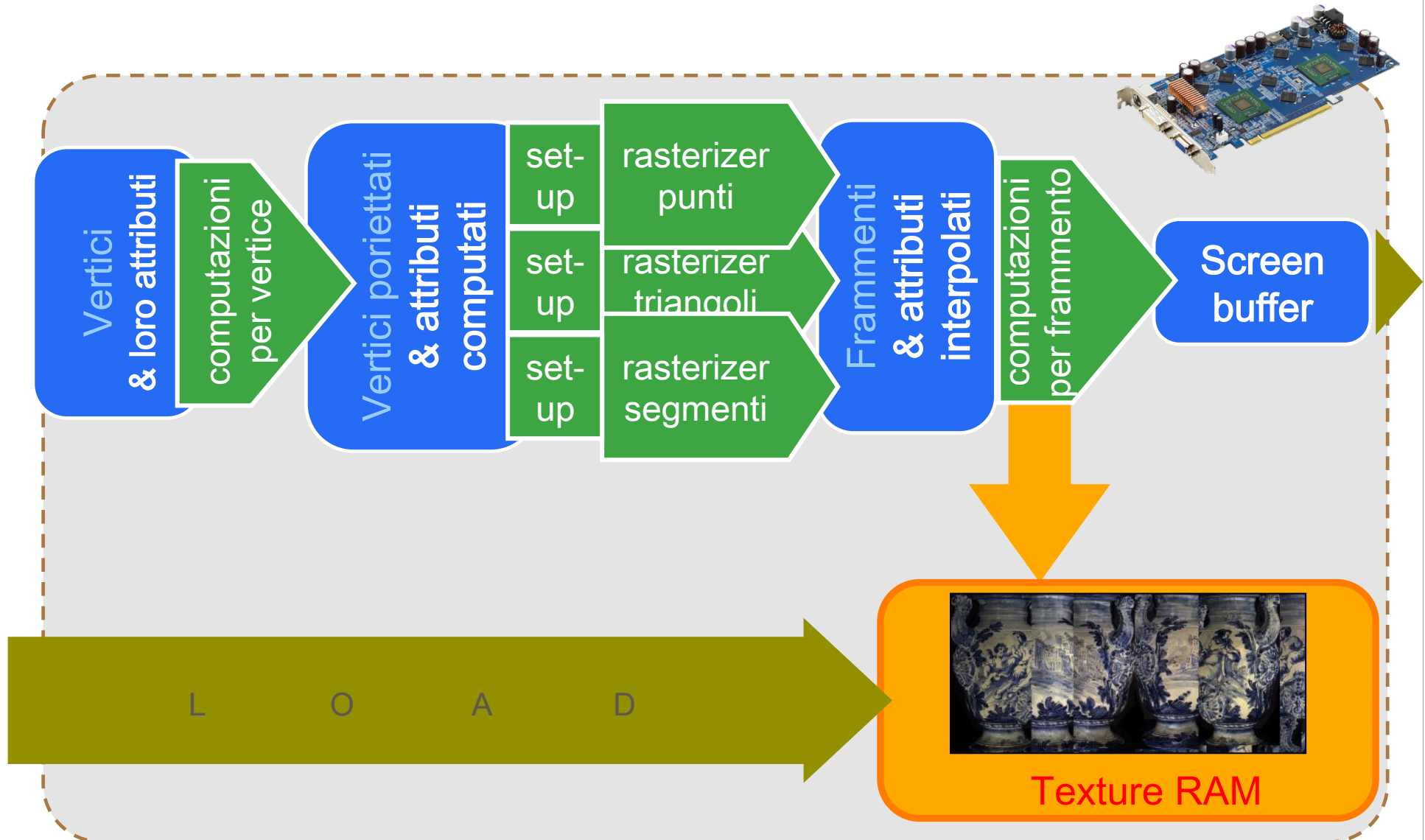
❖ Senza



❖ Con



Nota: la tessitura va caricata



Nota: la tessitura va caricata

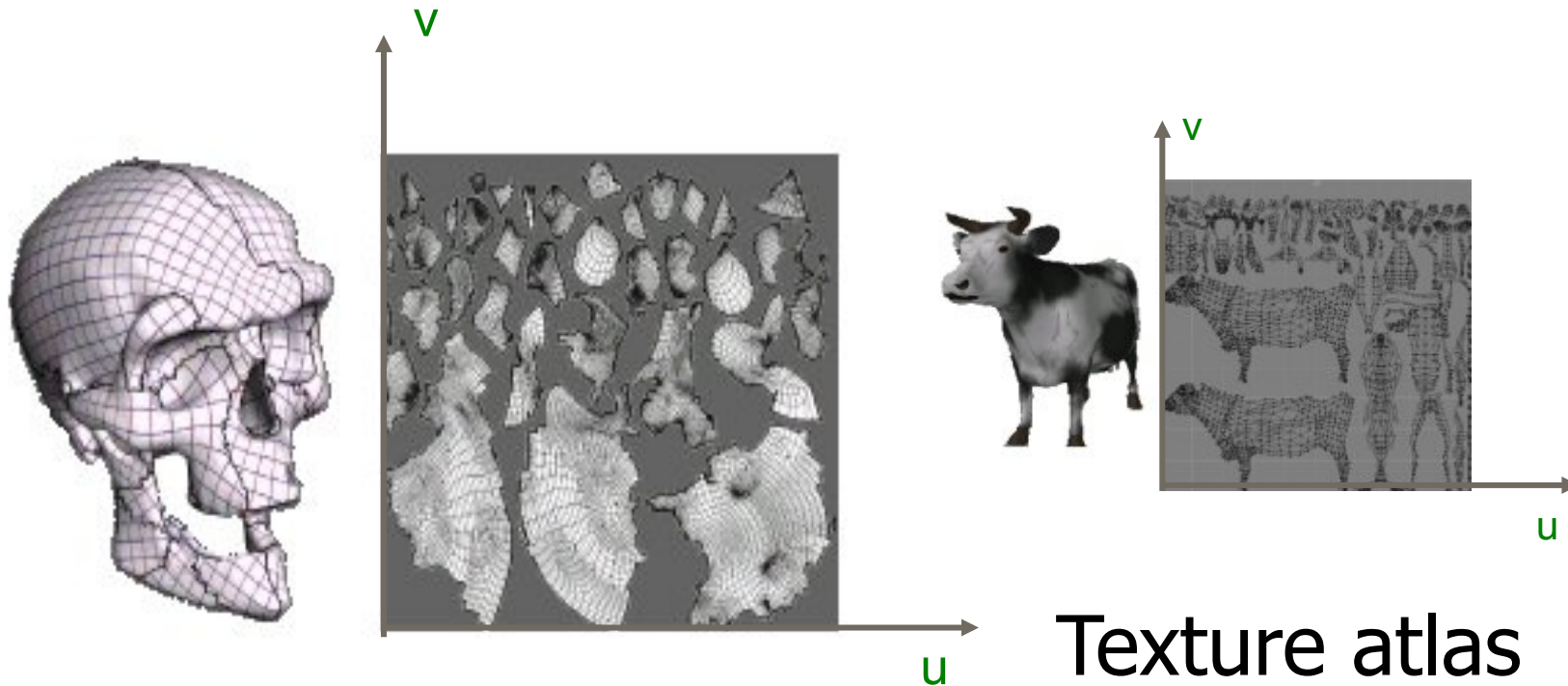
- ❖ Da disco a memoria RAM main (sulla scheda madre)
- ❖ Da memoria RAM main a Texture RAM (on board dell'HW grafico)
- ❖ Entrambe le operazioni sono piuttosto lente e sono impossibili da fare una volta per frame quindi nel progetto dell'applicazione si devono utilizzare strategie per la gestione delle texture

Assegnazione delle coordinate texture

- ❖ Due classi di soluzioni:
 - ❖ Calcolare le coordinate textures on-the-fly durante il rendering...
 - ❖ Precomputarle (e salvarle insieme alla mesh)
- ❖ Non esiste una soluzione ideale, dipende dall'applicazione che stiamo progettando
- ❖ Modelli con una sola texture l'avranno precomputata, per altri che variano dinamicamente l'assegneremo in rendering

Problema difficile: u-v mapping

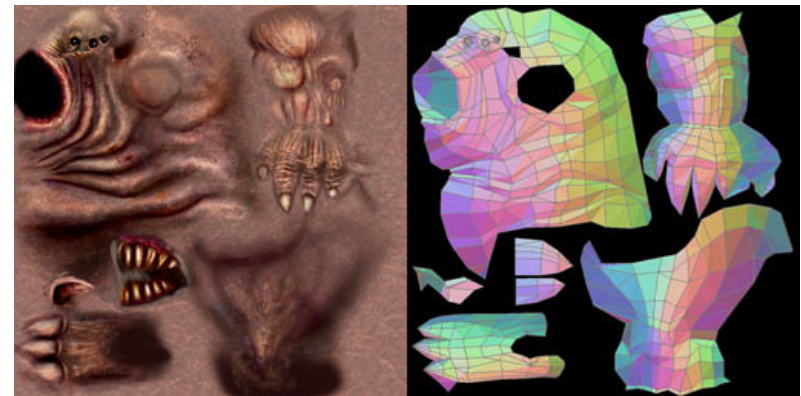
- ❖ Assegnare una coppia di coordinate textures ad ogni vertice della mesh in preprocessing



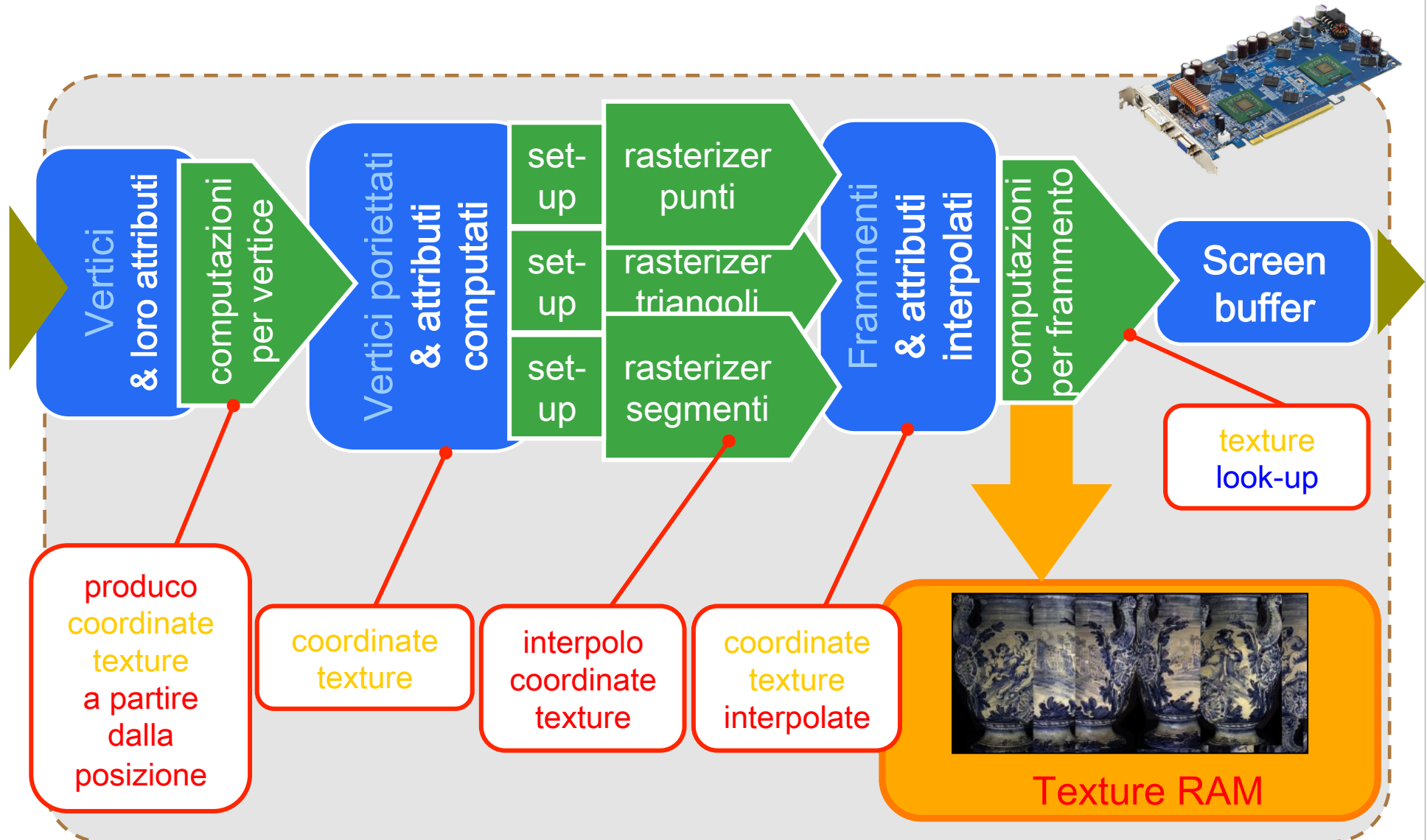
Problema difficile: u-v mapping



fatto a mano,
o automatizzato

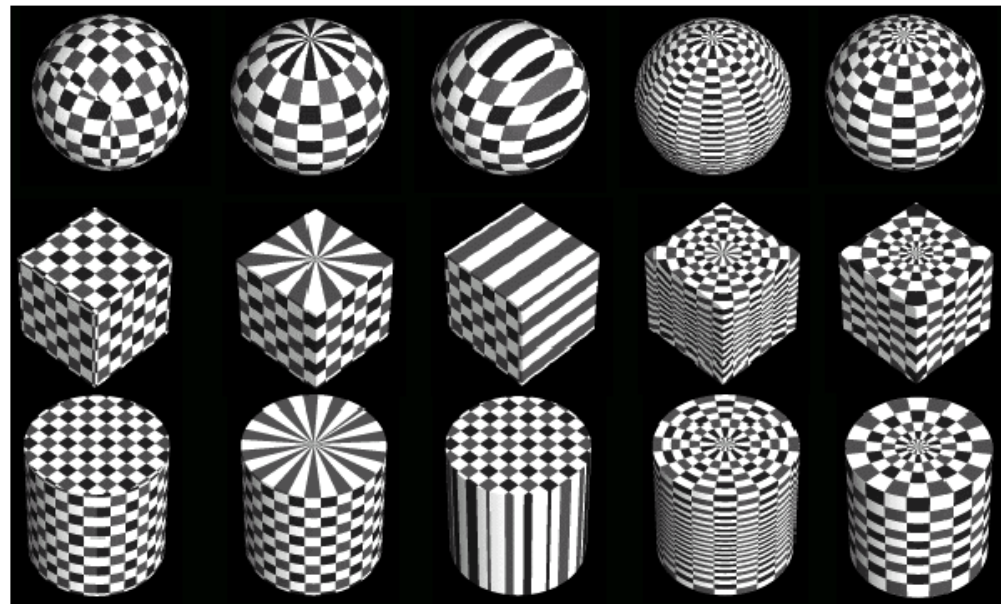


Come si assegnano le coordinate texture ai vertici?



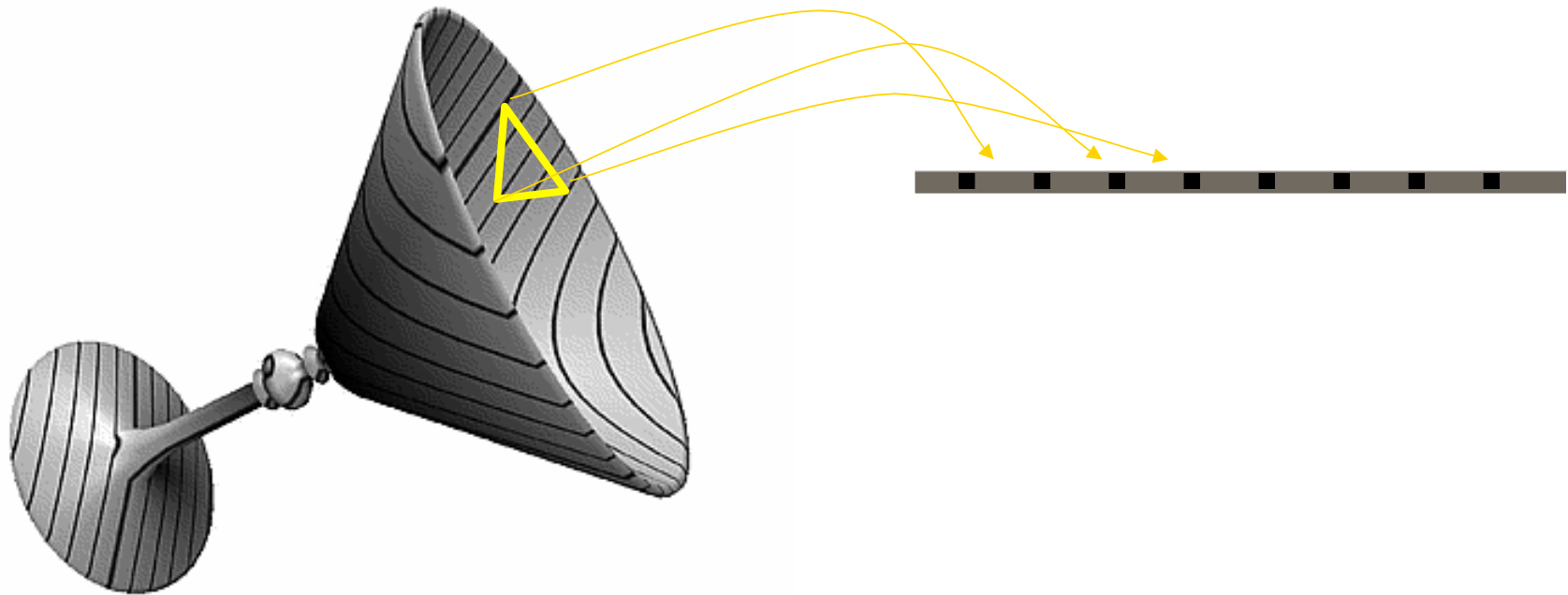
Create automaticamente

- ❖ Si utilizza un mapping lineare da (x,y,z) a (u,v) in coordinate oggetto o vista (prima o dopo la trasformazione)
- ❖ Esempi:



Create automaticamente

- ❖ La texture può anche essere una semplice immagine 1D

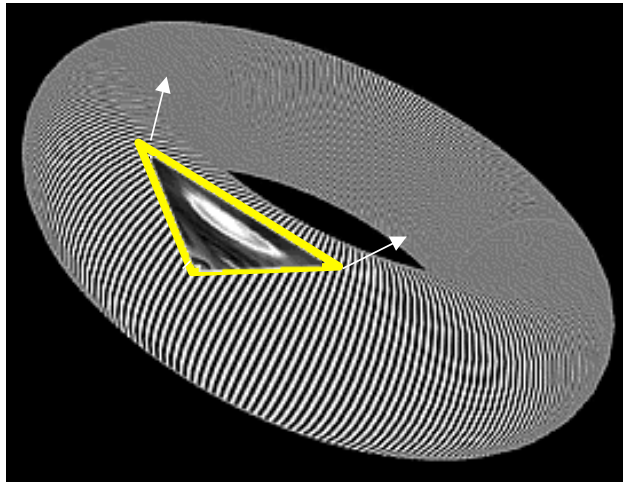


Environment mapping: sferico



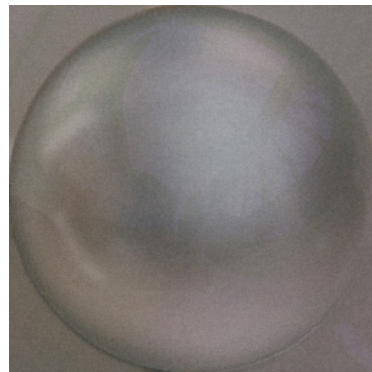
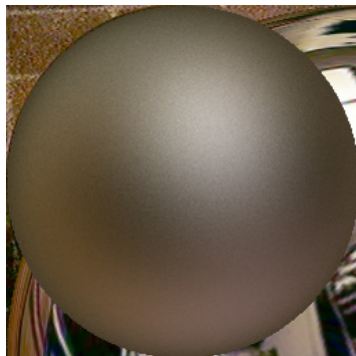
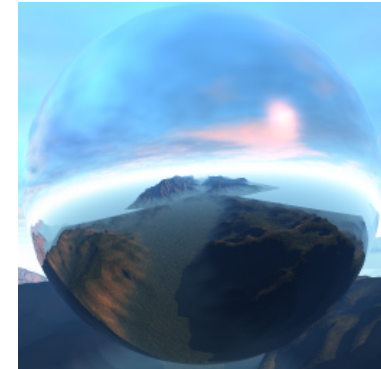
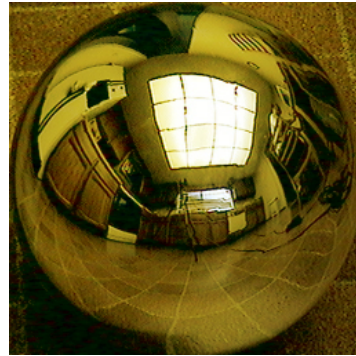
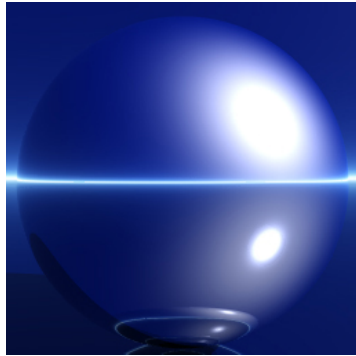
Environment map: una tessitura che memorizza il colore dell'ambiente "riflesso" da ogni normale della semisfera.

Come coordinata tessitura, basta usare la normale trasformata!

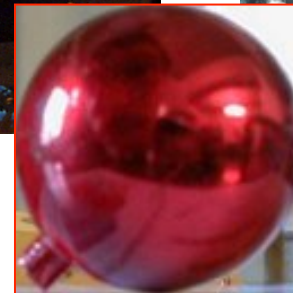


Environment mapping: sferico

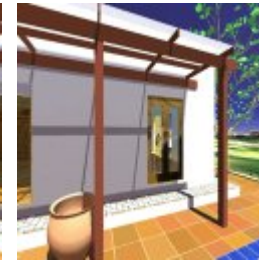
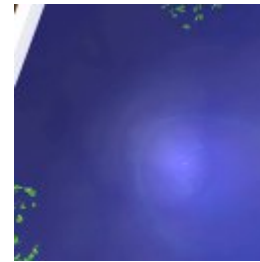
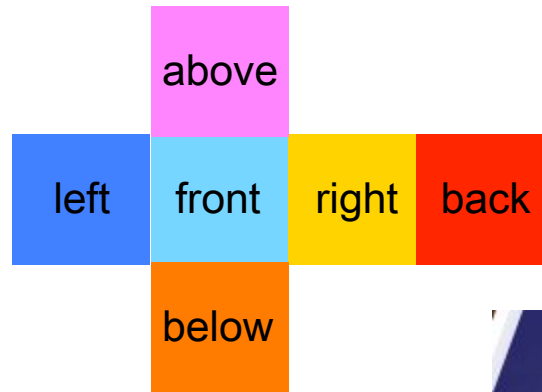
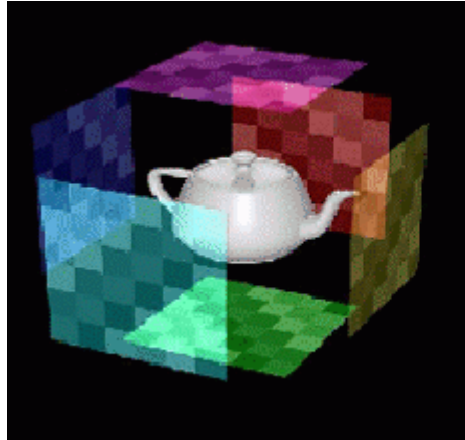
simula oggetto a specchio che riflette uno sfondo lontano



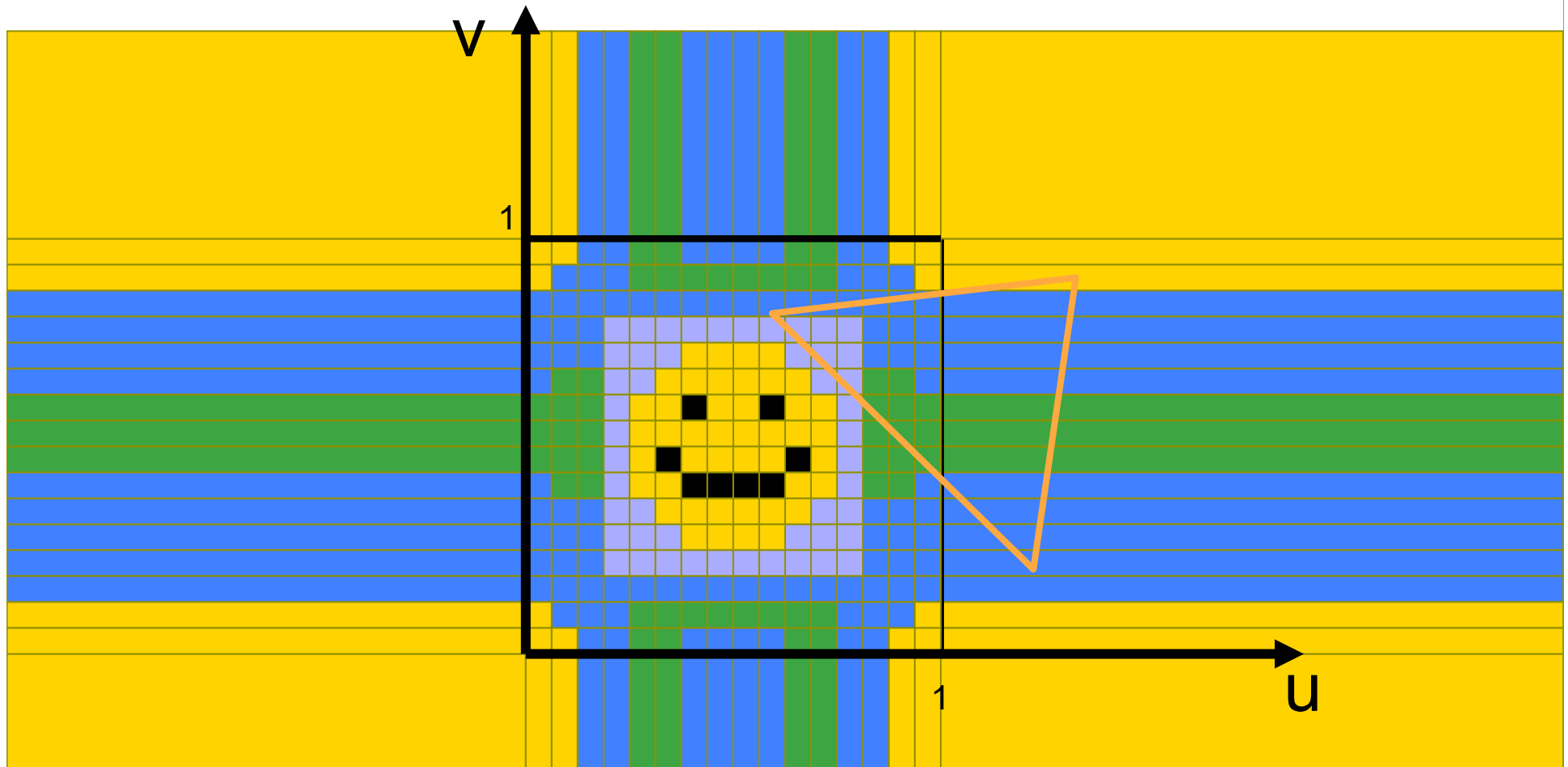
simula un materiale complesso
(condizioni di luce fisse)



Environment mapping: cubico

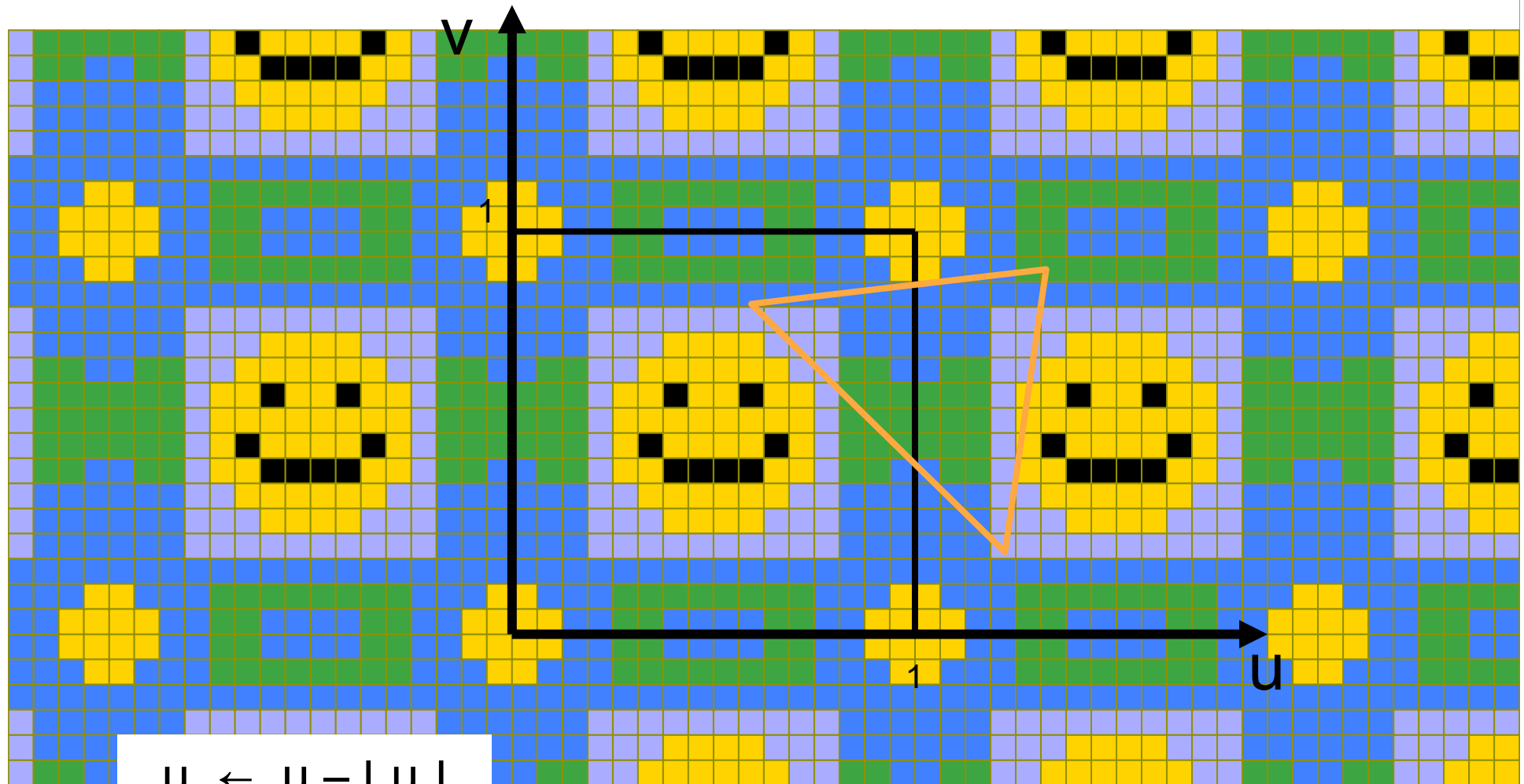


Texture fuori dai bordi: modo *clamp*



```
if (u < 0) u ← 0; if (u > 1) u ← 1;  
if (v < 0) v ← 0; if (v > 1) v ← 1;
```


Texture fuori dai bordi: modo *repeat*

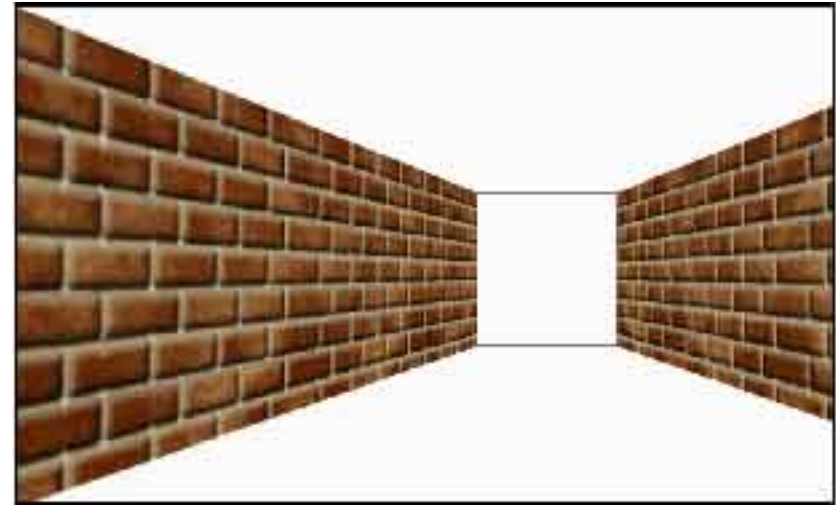
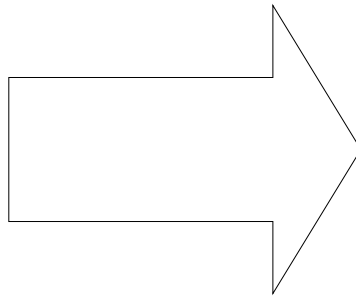


$$u \leftarrow u - [u]$$
$$v \leftarrow v - [v]$$

Tessiture ripetute

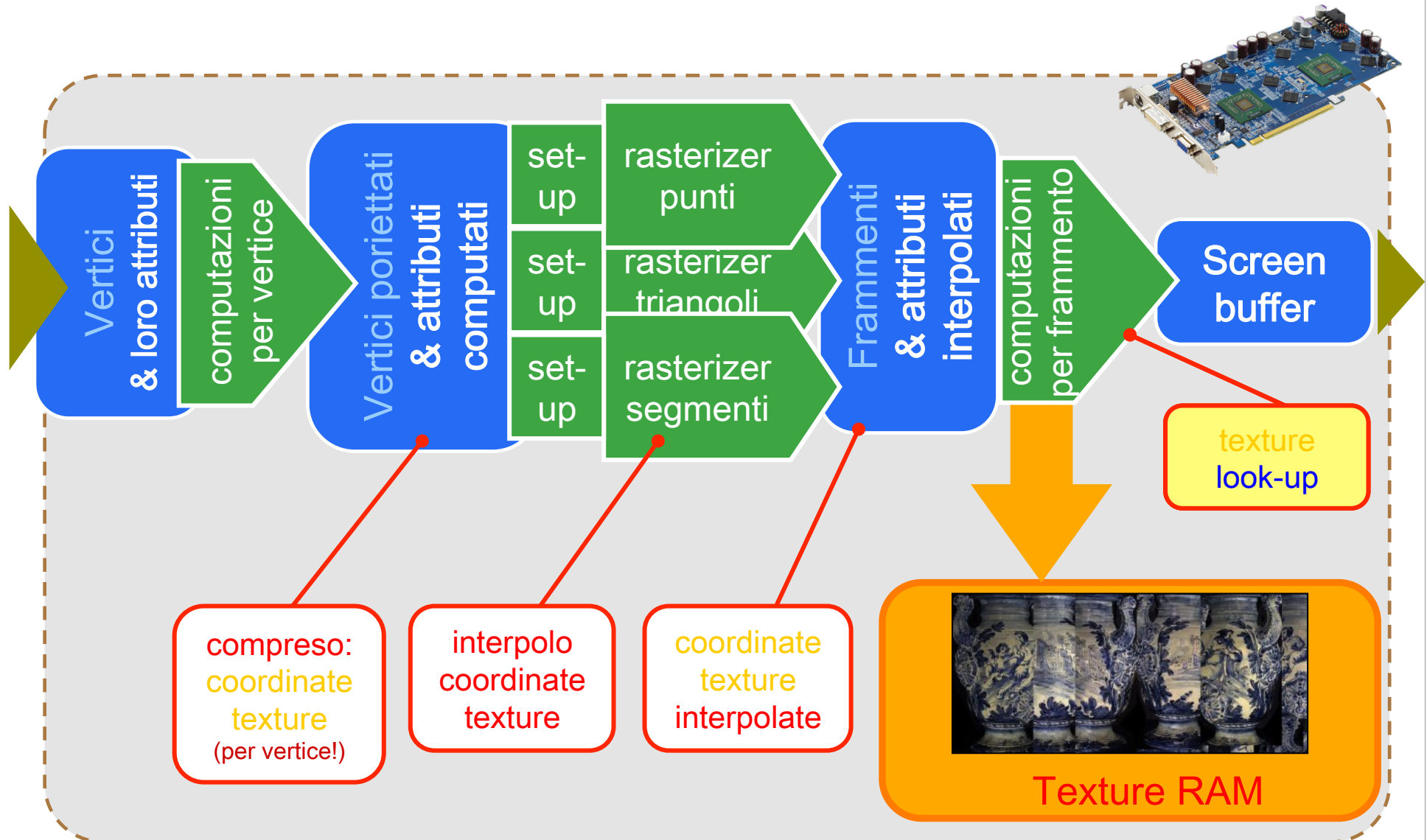
❖ Tipico utilizzo:

Nota: deve essere **TILABLE**



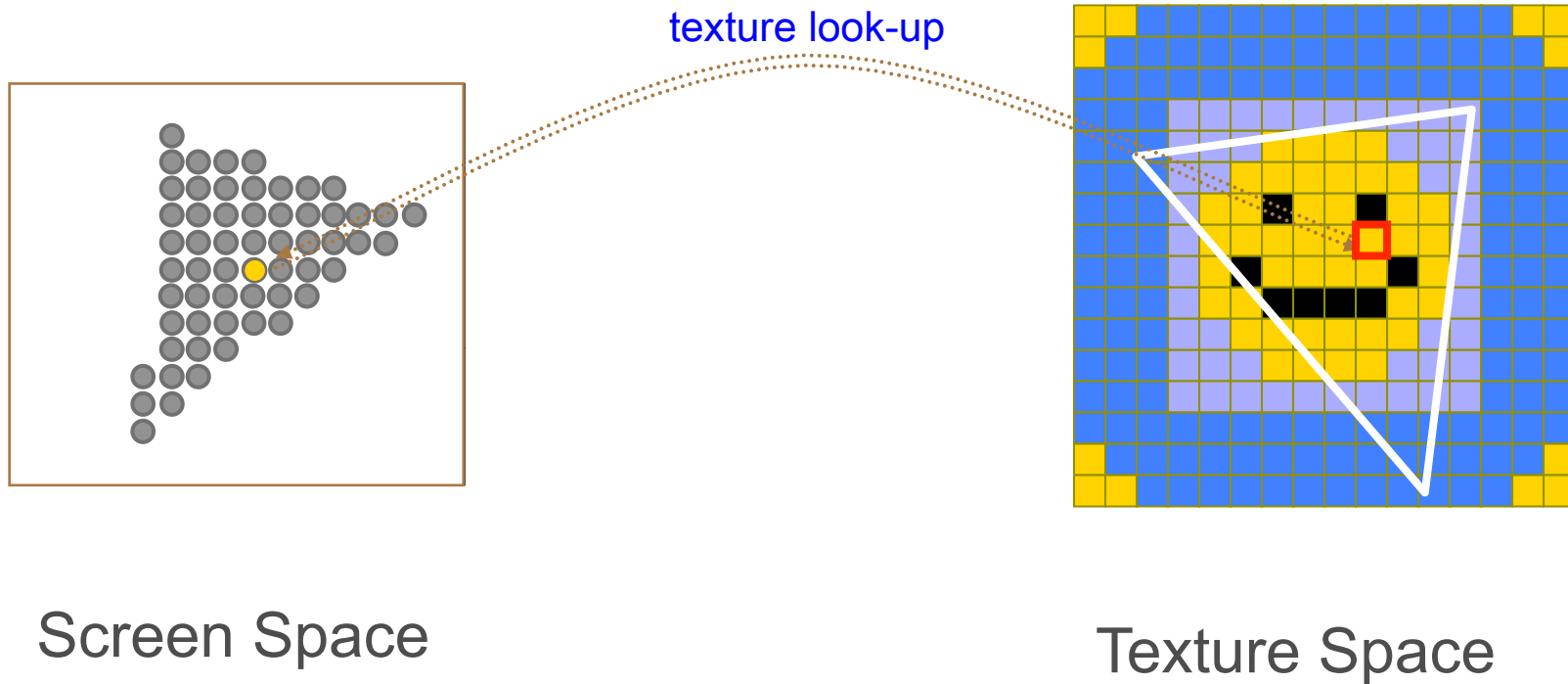
Molto efficiente in spazio: una sola texture mappa su molti triangoli

Texture Mapping

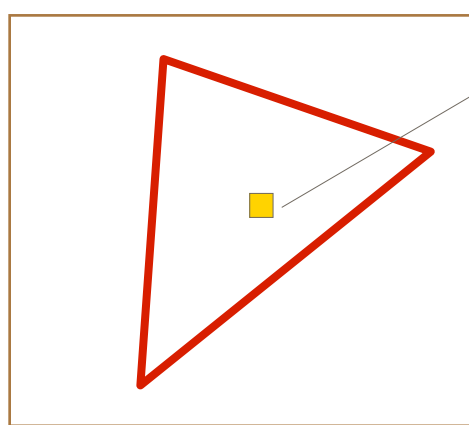


Texture Look-up

- ❖ Un frammento ha coordinate non intere (in texels)



Texture Look-up

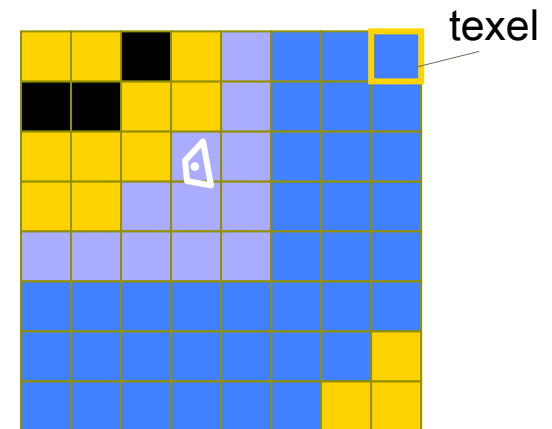


pixel

un pixel = meno di un texel

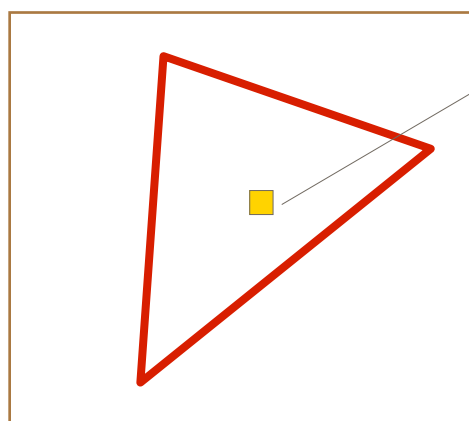
magnification

Screen Space



texel

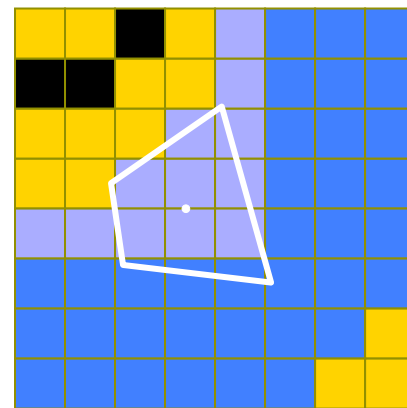
Texture Space



pixel

un pixel = più di un texel

minification



Caso Magnification

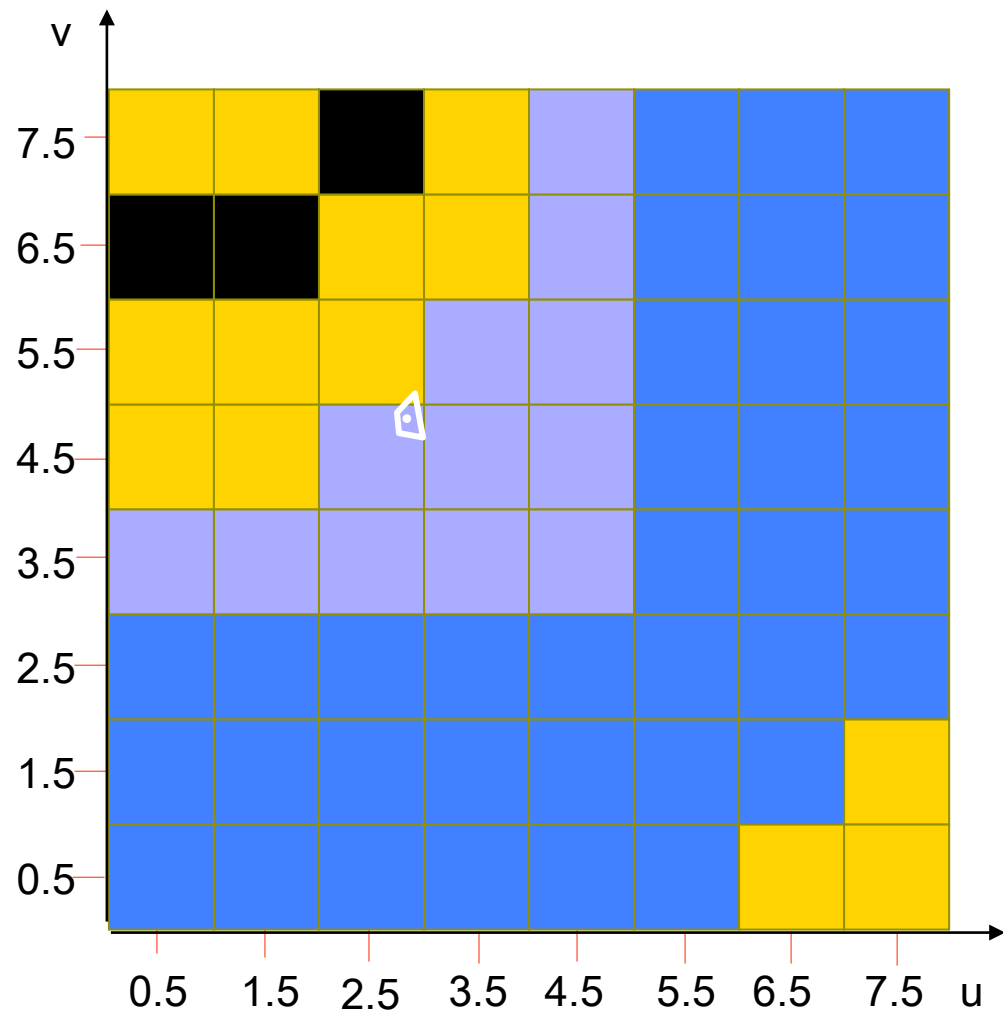
Soluzione 1:

prendo il texel in cui sono

(equivale a prendere
il texel più vicino)

equivale ad arrotondare
alle coordinate texel
interi

"Nearest Filtering"



Caso Magnification

Nearest Filtering: risultato visivo



texture 128x128

"si vedono i texel !"

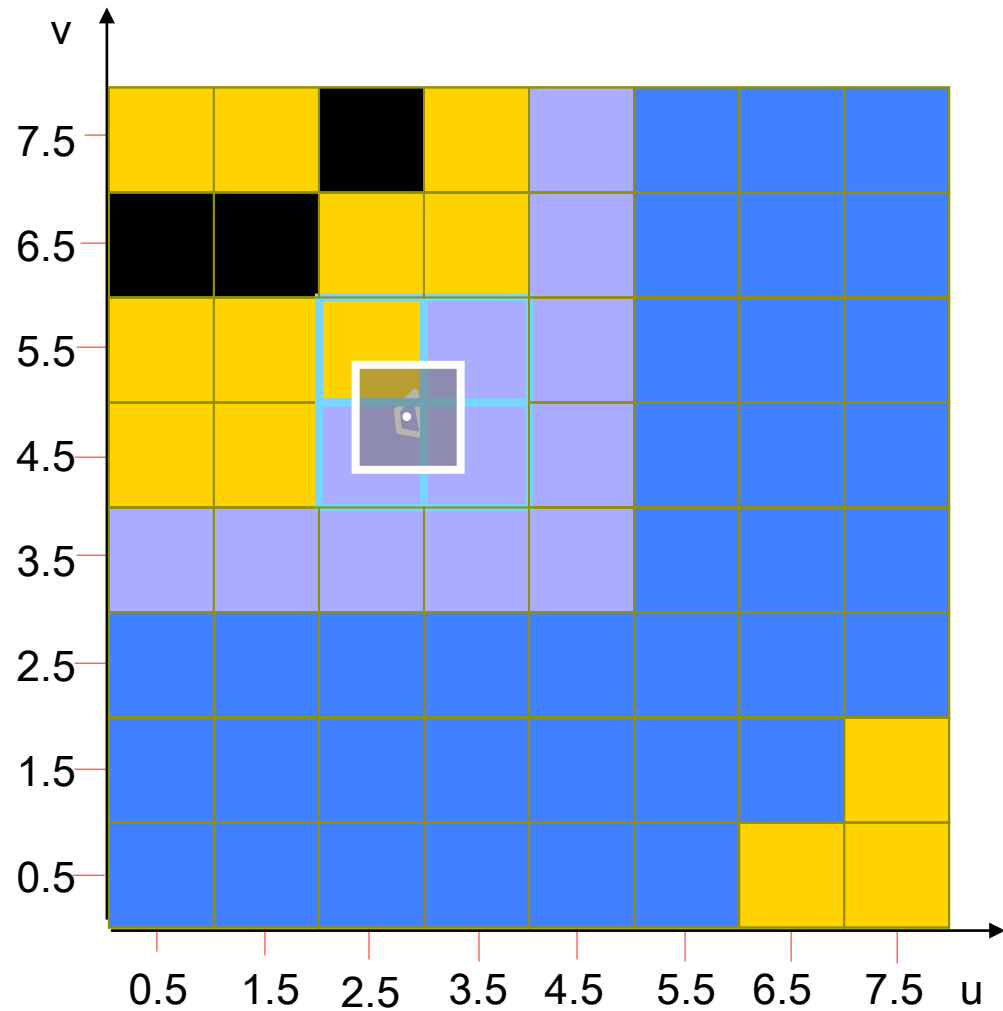


Caso Magnification

Soluzione 2:

Medio il valore dei quattro texel più vicini

Interpolazione Bilineare



Caso Magnification

Bilinear Interpolation: risultato visivo



texture 128x128



Caso Magnification

❖ Modo Nearest:

- ❖ si vedono i texel

- ❖ va bene se i bordi fra i texel sono utili

- ❖ più veloce

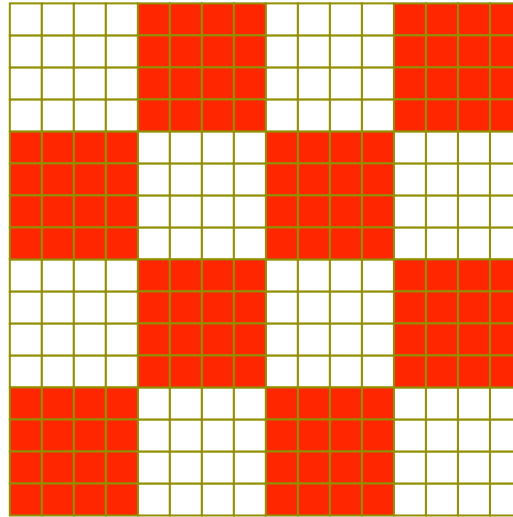
❖ Modo Interpolazione Bilineare

- ❖ di solito qualità migliore

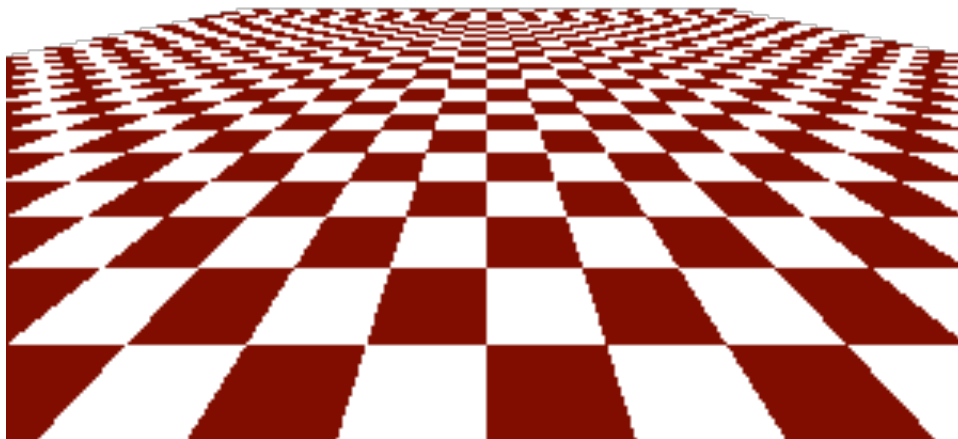
- ❖ può essere più lento

- ❖ rischia di avere un effetto "sfuocato"

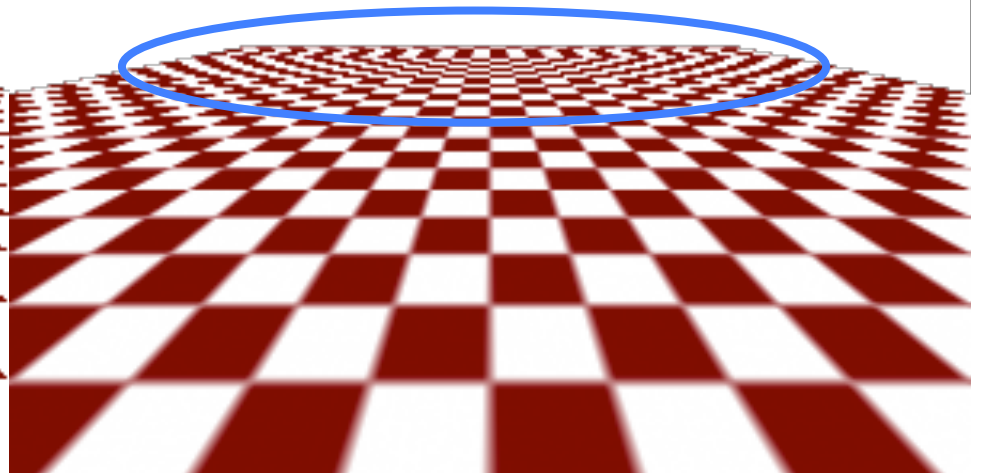
Caso Minification



Nearest Filtering

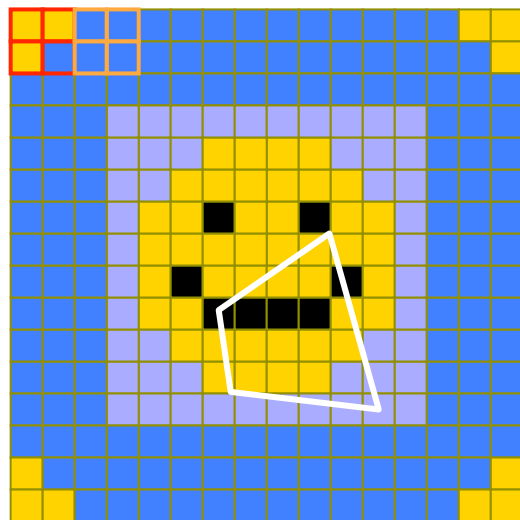


Bilinear interpolation
non risolve il problema

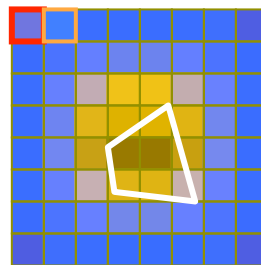


Caso Minification: MIP-mapping

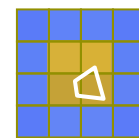
MIP-mapping: "Multum In Parvo"



MIP-map
level 0



MIP-map
level 1



MIP-map
level 2



MIP-map
level 3

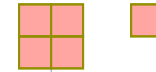
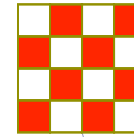
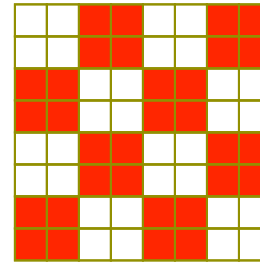
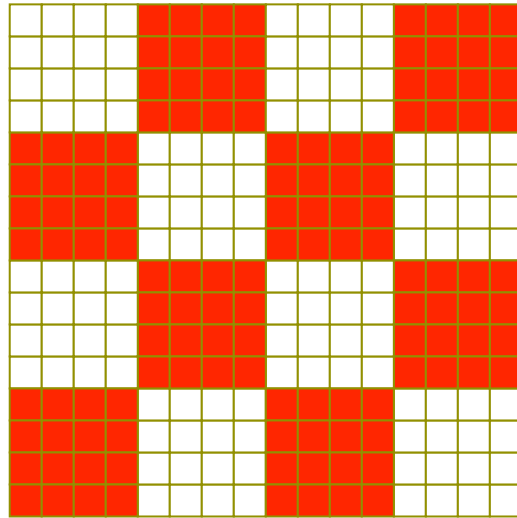


MIP-map
level 4
(un solo texel)

Mipmap Math

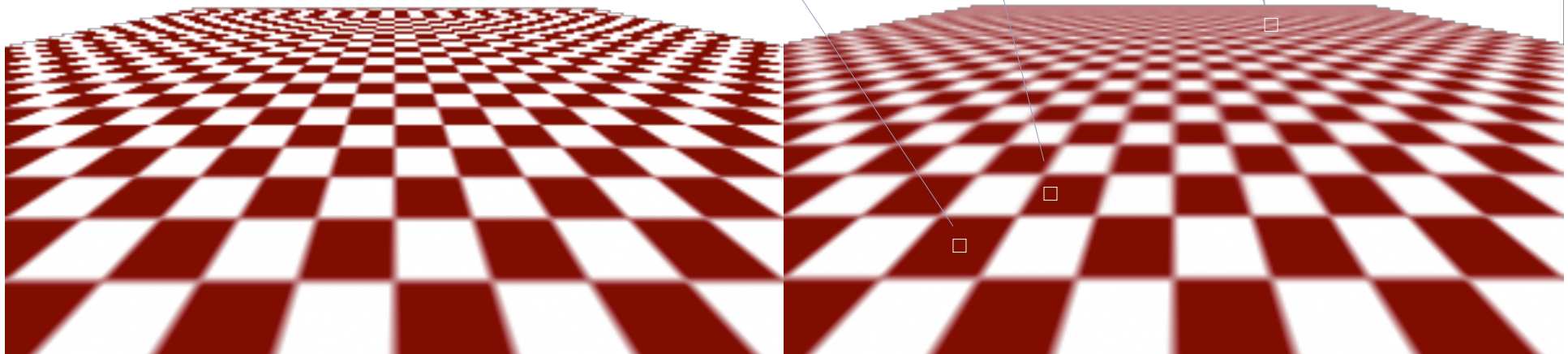
- ❖ Definiamo un **fattore di scala**, $\rho = \text{texels/pixel}$ come valore massimo fra ρ_x e ρ_y , che può variare sullo stesso triangolo, può essere derivato dalle matrici di trasformazione ed è calcolato nei **vertici**, interpolato nei **frammenti**
- ❖ Il **livello di mipmap** da utilizzare è: $\log_2 \rho$ dove il livello 0 indica la massima risoluzione
- ❖ Il livello non è necessariamente un numero intero e può quindi essere arrotondato

Caso Minification: MIP-mapping

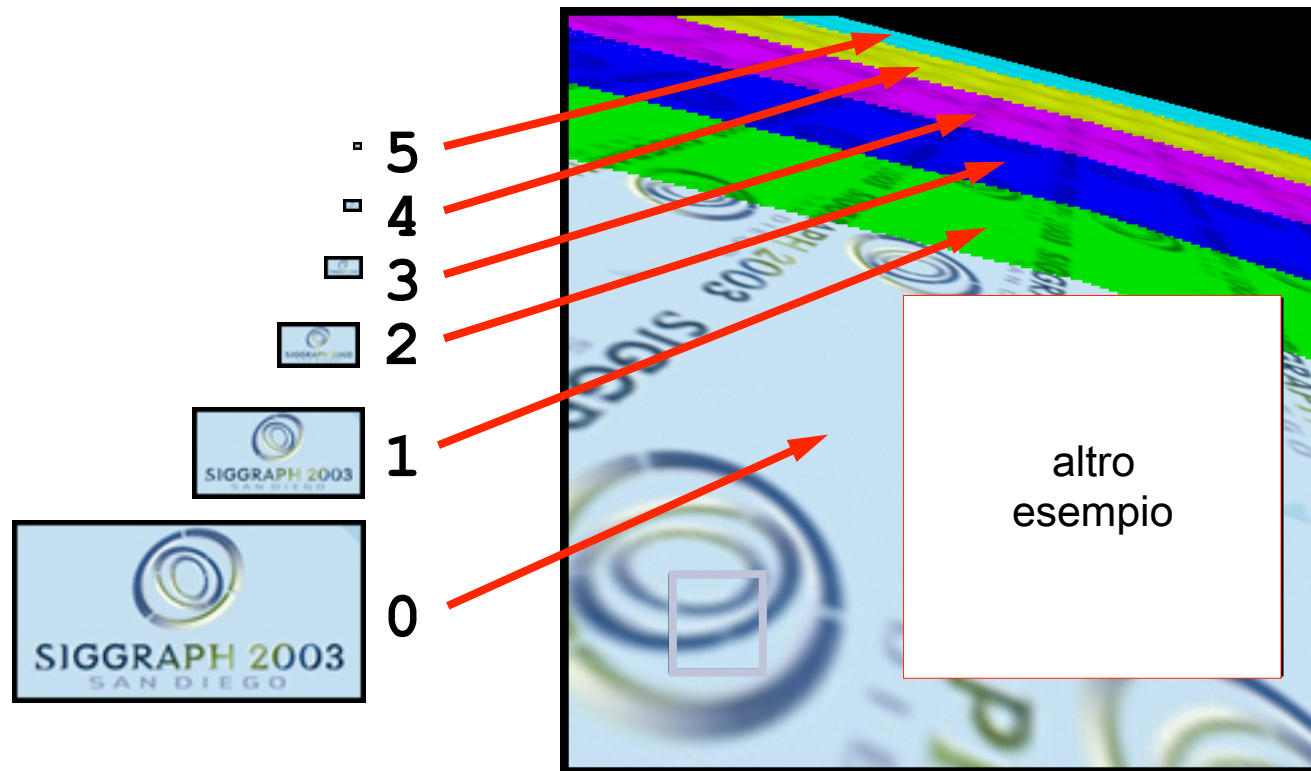


Bilinear interpolation
non risolve il problema

MIP-mapping

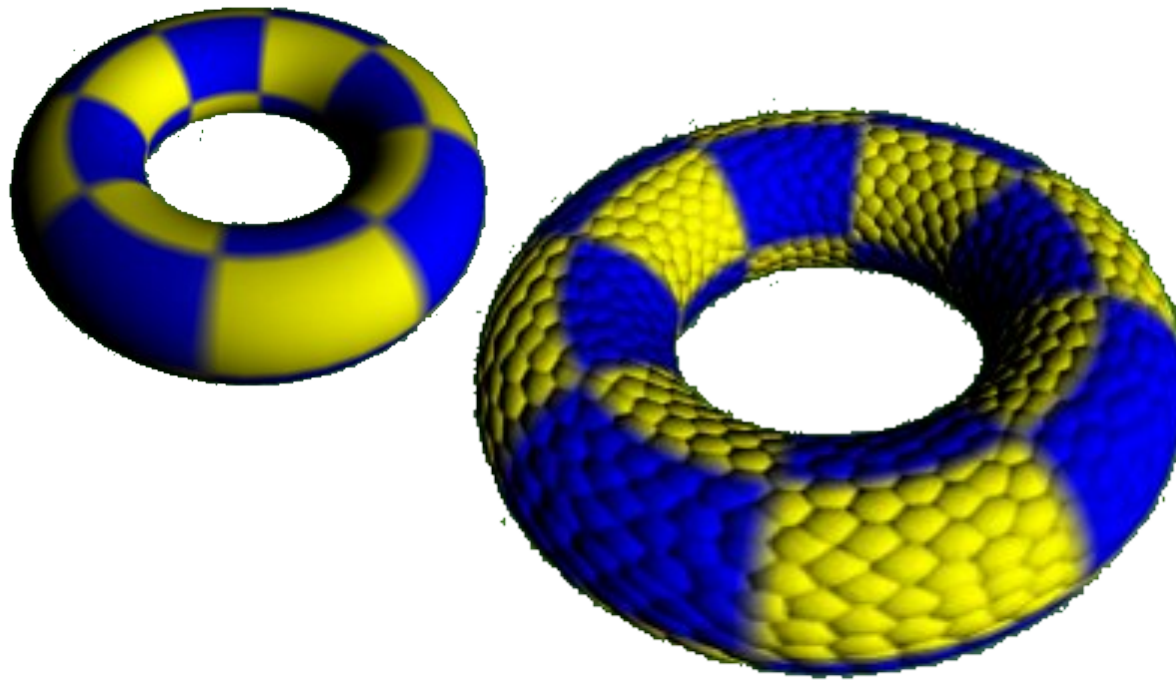


Caso Minification: MIP-mapping



Bump mapping

- ❖ Un'ulteriore modifica all'apparenza del rendering può essere effettuata usando il bump mapping



Bump mapping

- ❖ Il metodo prevede di variare la normale alla superficie pixel per pixel utilizzando la formula:

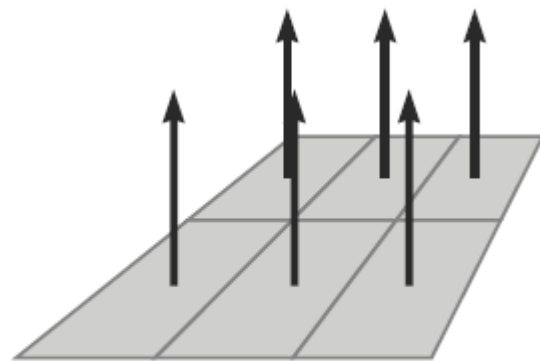
$$\vec{N}_{new} = \vec{N}_{old} + \vec{D};$$

$$\vec{D} = (\Delta x, \Delta y, \Delta z)$$

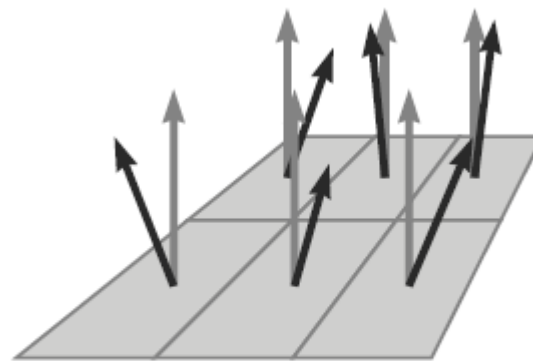
- ❖ I texel in questo caso sono utilizzati ad uno stadio diverso rispetto ai color texel, prima del calcolo dell'equazione di illuminazione

Bump mapping

- ❖ L'effetto che si ottiene è un perturbazione del valore delle normali che altera il rendering senza modificare la geometria

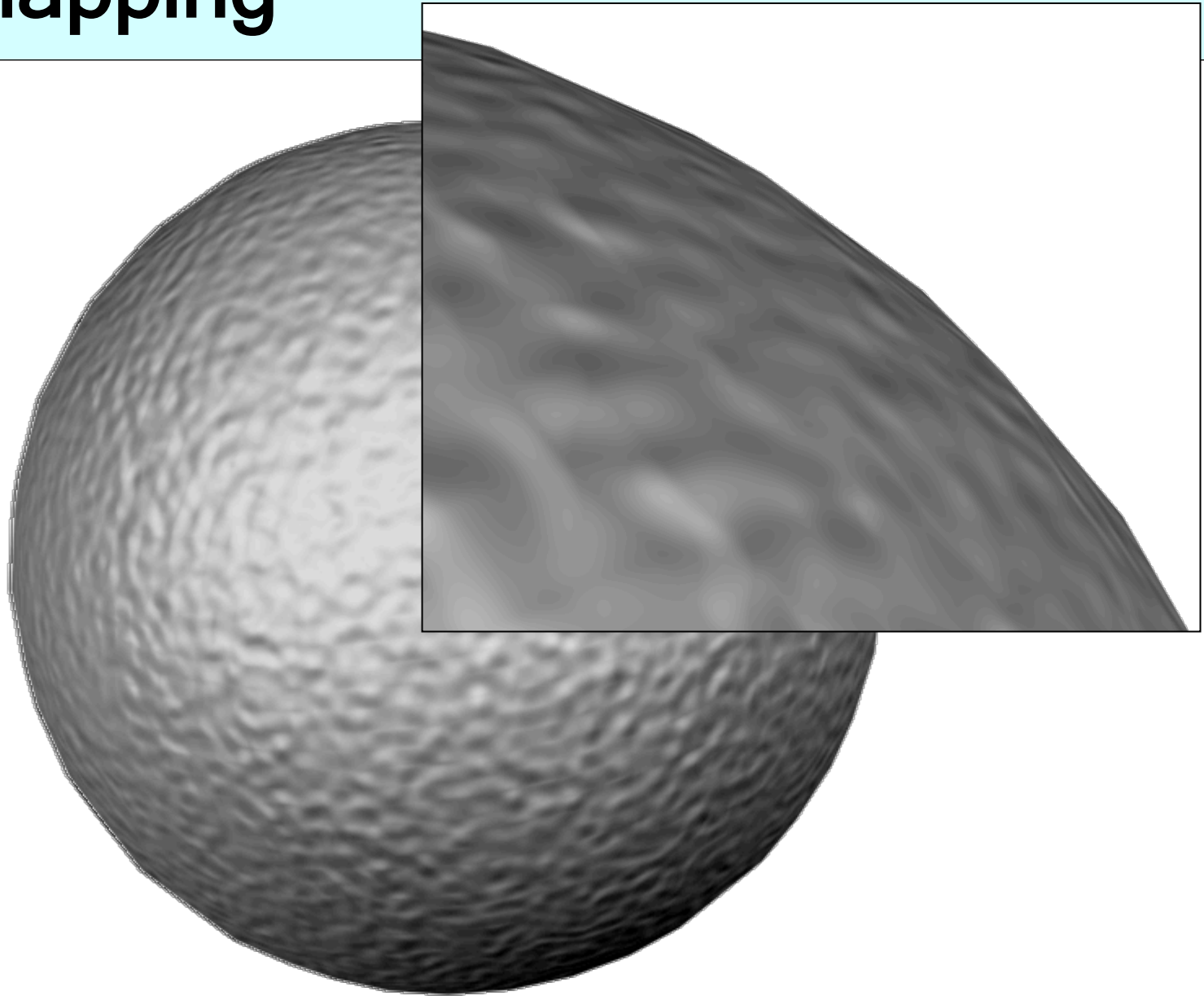


Superficie originale



Nuove normali

Bump mapping



Displacement mapping

- ❖ Nel displacement mapping si modifica effettivamente la geometria dell'oggetto spostando i punti della superficie:

$$P_{new} = P_{old} + h \cdot \vec{N}$$

- ❖ Il displacement mapping è eseguito in fase di rendering e non modifica stabilmente la geometria della scena
- ❖ Rispetto al bump mapping anche la silhouette del modello mostra le corrette deformazioni

Displacement mapping

