Tecniche di mappatura

Dove si discute l'utile tecnica di texture mapping che consente di aggiungere dettagli alle superfici senza gravare sulla geometria.

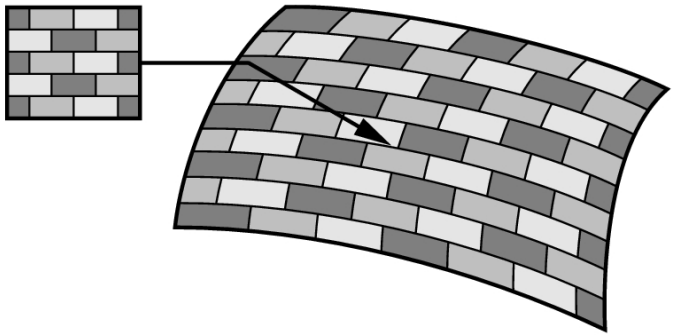
- **Introduzione**
- **Texture mapping**
- **Bump mapping**

Introduzione

- Il modello di illuminazione di Phong è abbastanza versatile: con una scelta opportuna dei vari parametri si possono imitare diversi materiali in modo abbastanza realistico (per esempio la plastica)
- È comunque limitato: non si possono simulare i dettagli di un materiale, a meno di non introdurli nella geometria (ovvero aumentare il numero dei poligoni).
- Invece di incrementare la complesssità del modello, si aggiungono particolari come parte del processo di rendering.
- Le tecniche di mappatura (texture mapping) interagiscono con lo shading usando una mappa bidimensionale o **texture** (tessitura) come tabella di lookup, in modo da aggiungere dettagli alla superficie.
- Si possono ottenere risultati ottimi dal punto di vista del fotorealismo con un limitato carico computazionale.
- Il texture mapping è ampliamente sopportato dalle schede grafiche.

Texture mapping

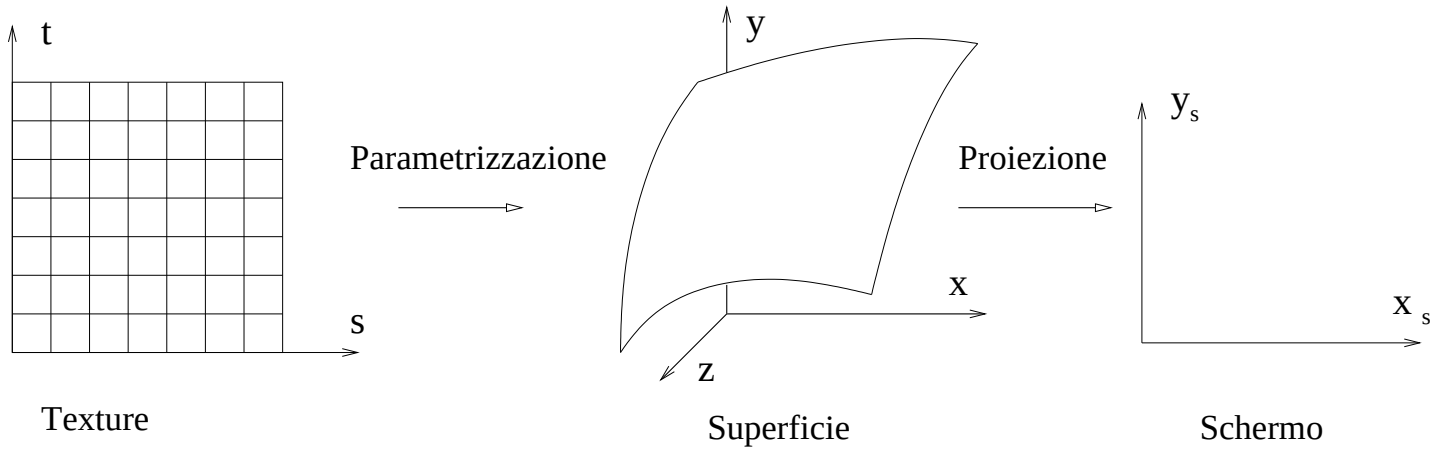
- Il texture mappping, nella sua forma più semplice (color mapping) consiste nell'applicare una immagine su una superficie, come una decalcomania.
- Esempi: una etichetta su una lattina, una foto su un cartellone pubblicitario, oppure tessiture regolari come legno o marmo su una superficie.



©E. Angel

- Una texture map è una matrice bidimensionale di dati indirizzati da due coordinate s e t comprese tra 0 ed 1. Il dato contenuto è tipicamente un colore (la mappa è una immagine), ma potrebbe essere qualcos'altro.
- Più in generale una texture map può contenere qualunque tipo di informazione che incide sull'apparenza di una superficie: la texture map è una tabella ed il texture mapping consiste nel recuperare dalla tabella (lookup) l'informazione che serve per effettuare il rendering di un certo punto.

- La texture può essere applicata dopo il calcolo della illuminazione con Phong (per modificare attributi come il colore, la luminosità o la trasparenza) oppure può modificare i parametri (come le normali) che entrano nel modello di Phong.
- Un passaggio chiave è stabilire una corrispondenza univoca tra superficie dell'oggetto e texture.
- Occorre definire la funzione di **parametrizzazione** $W()$ che associa un punto (s,t) della texture ad un punto P della superficie dell'oggetto 3D (è una funzione che "spalma " la texture sulla superficie).
- Il punto P viene poi mappato dalla proiezione in un punto (x_s, y_s) dello schermo.



- il **rendering** della texture si occupa poi di stabilitre il valore di texture da associare a ciascun pixel.

Parametrizzazione

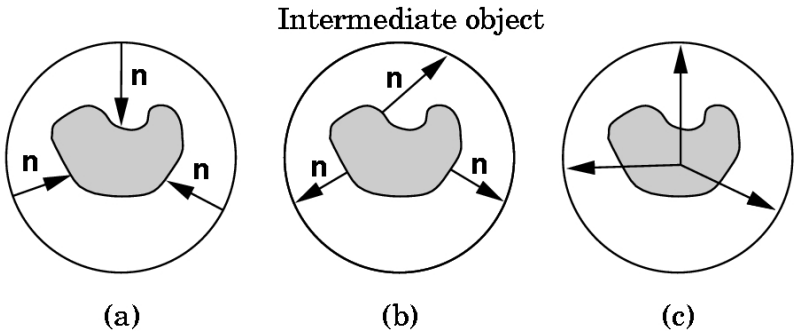
- Nella mappatura **in un passo** si definisce (in forma analitica) la funzione W che definisce la corrispondenza tra i pixel della texture ed i punti della superficie.
- Questo si può fare quando si ha la descrizione parametrica della superficie soggiacente alla maglia poligonale.
- Altrimenti si specifica tabularmente la corrispondenza W^{-1} tra vertici della maglia e punti della texture.
- Se la superficie è data in forma parametrica, ad ogni suo punto P sono associate due coordinate (parametri) (u, v).
- Per ottenere W basta specificare la mappa che va da (u, v) sulla superficie a (s, t) nella texture.
- è opportuno che W sia invertibile.
- Spesso è l'identità (con qualche fattore di normalizzazione).
- Ad esempio si consideri un cilindro di altezza h. per il quale si definisce la funzione

$$W : (\theta, z) \rightarrow (s, t) = (\frac{m}{2\pi}\theta, \frac{n}{h}z)$$

dove (m, n) sono le dimensioni della texture map.

- Una tecnica più generale, che si può usare senza conoscere l'equazione parametrica della superficie, è la mappatura **in due passi**
- Si mappa la texture su una superficie intermedia semplice, in modo che la parametrizzazione (corrispondenza punti-superficie con pixel-texture) sia immediata; questa prende il nome di **S-mapping**
- Quindi si mappa ogni punto della superficie intermedia in un punto della superficie in esame; questa prende il nome di **O-mapping**
- La concatenazione dei due mapping genera la corrispondenza W trai pixel della texture ed i punti dell'oggetto
- Il primo passaggio (S-mapping) è in genere semplice; basta scegliere superfici facili da parametrizzare
- Ad esempio si può prendere come superficie intermedia un cilindro (vedi slide precedente)
- Oltre al cilindro è facile fare l'S-mapping con cubi, piani e sfere.
- In genere si considera la superficie intermedia come esterna all'oggetto da tessiturare.

- Per l'O-mapping ci sono varie scelte
  1. Si considera la normale uscente da un punto dell'oggetto; il raggio che passa per tale punto e con direzione tale normale intersecherà la superficie intermedia in un punto, stabilendo così l'O-mapping
  2. Anziché usare la normale si può usare la retta che congiunge il centroide dell'oggetto con il punto considerato
  3. Altrimenti si può considerare la normale in un punto della superficie intermedia e la retta che passa per tale punto e con direzione questa normale, intersecherà la superficie dell'oggetto in un punto, stabilendo un altro possibile O-mapping



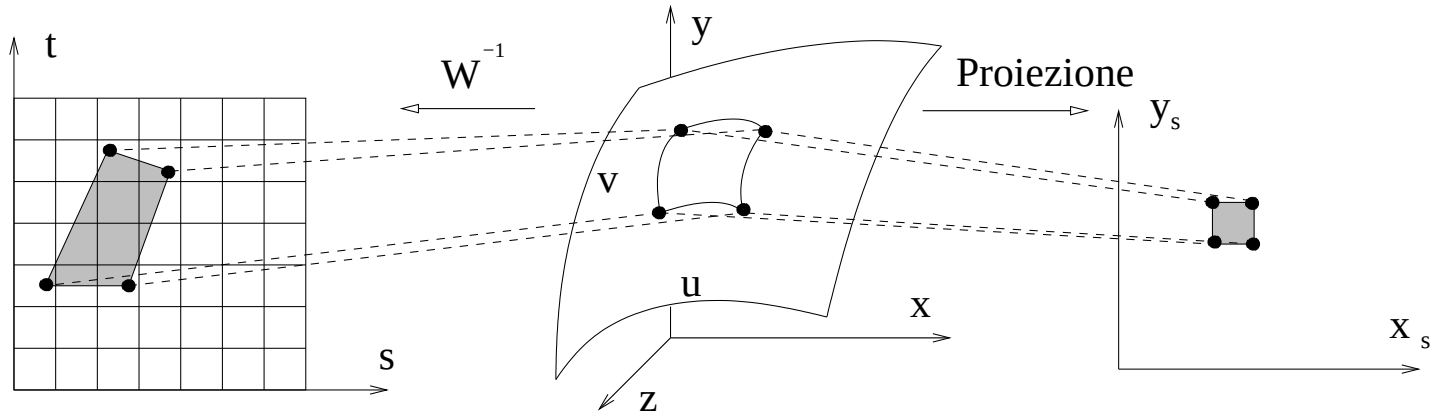
©E. Angel

Esempi di O-mapping. (a) Usando la normale alla superficie intermedia. (b) Usando la normale dalla superficie dell'oggetto. (c) Usando i raggi dal centro dell'oggetto.

Rendering

- Quello che serve per il rendering è la corrispondenza tra texture (s, t) e pixel dell'immagine (x_s, y_s).
- Vi sono tipicamente due strategie per poter mappare una texture:
  1. **Mappatura in avanti:** (forward mapping) dato un pixel (s, t) nella texture si trova il punto (x, y, z) dell'oggetto su cui tale pixel viene mappato da W e quindi, con la normale proiezione che abbiamo studiato, si trova il suo corrispondente (x_s, y_s).
  2. **Mappatura all'indietro:** (inverse mapping) Per ogni pixel (x_s, y_s) si trova nella texture la corrispondente coordinata (s, t). Si usa la retroproiezione del pixel sulla superficie e la funzione W^{-1}.
- Per evitare fenomeni di aliasing, nell'assegnare un valore di texture ad un pixel dell'immagine bisogna tenere presente che la "pre-immagine" di un pixel nello spazio texture non è un punto, ma un quadrilatero curvilineo (in generale), il quale può comprendere al suo interno molti pixel della texture.
- Si usa quasi sempre l'inverse mapping.
- Vediamo ora due esempi di mappatura inversa, uno per superfici parametriche, l'altro per maglie poligonali.

- Supponiamo che siano date la funzione W e l'equazione parametrica della superficie.
- Dato il pixel che vogliamo disegnare, lo si (retro)proietta sulla superficie (Si usano i quattro vertici). La sua "impronta" sulla superficie è un patch quadrangolare.

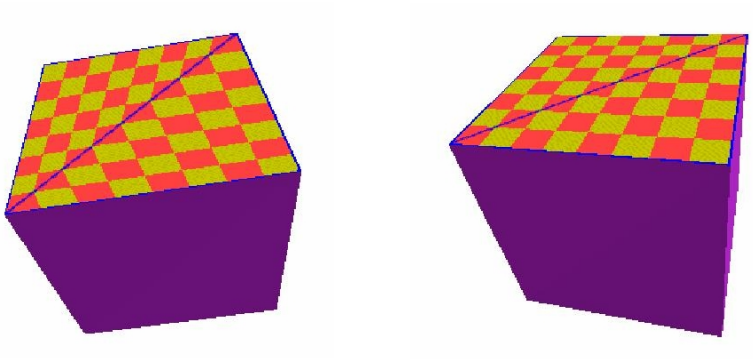


- Questo patch è descritto da due parametri (u, v) che, mappati con W^{-1} in (s, t), definiscono una quadrilatero nella texture.
- In realtà l'immagine di un pixel (quadrato) secondo il la mappatura all'indietro è un quadrilatero curvilineo nello spazio texture.
- Integrando sul quadrilatero si ottiene il valore per il pixel.
- Questa può essere presa anche come descrizione del processo "ideale" di texture mapping.

- Vediamo ora un esempio (vicino a OpenGL) di come avviene il rendering della texture nel caso di maglia poligonale.
- La mappatura W^{-1}, associa una coordinata texture (s, t) ad ogni vertice della maglia;
- Quindi a ciascun vertice di un triangolo proiettato è associata una coordinata texture (s, t).
- Durante la scan conversion del triangolo si determinano le coordinate texture di ciascun pixel interno conl'interpolazione scan-line (come abbiamo già usato per la profondità e per lo shading di Gouraud).
- Si determina quindi il valore di texture da associare al pixel arrotondando le coordinate texture all'intero più vicino. In questo modo ogni pixel riceve contributo da un solo texel (texture element).
- Se la risoluzione della texture è molto diversa da quella del display (ovvero la corrispondenza pixel-texel è molti a uno o uno a molti) si incorre nell'**aliasing** (nelle prossime diapositive discuteremo meglio il problema).

Distorsione prospettica

- L'interpolazione scan-line (lineare) di (s, t) crea delle distorsioni prospettiche nell'immagine risultante (in effetti viene a mancare la corretta distorsione prospettica, o foreshortening). La soluzione è impiegare l'**interpolazione iperbolica**, che apporta l'opportuna correzione prospettica.
- In pratica, bisogna interpolare non (s, t), ma le coordinate omogenee $(u/z, v/z, 1/z)$ e poi determinare il pixel della texture dividendo per l'ultima componente.
- Nella figura sotto, per esempio, la faccia del quadrato è composta da due triangoli, ai quali viene applicata la stessa texture. Usando l'interpolazione scan-line senza correzione prospettica (sinistra) si nota un effetto indesiderato lungo la diagonale, che invece scompare usando l'interpolazione iperbolica (destra).

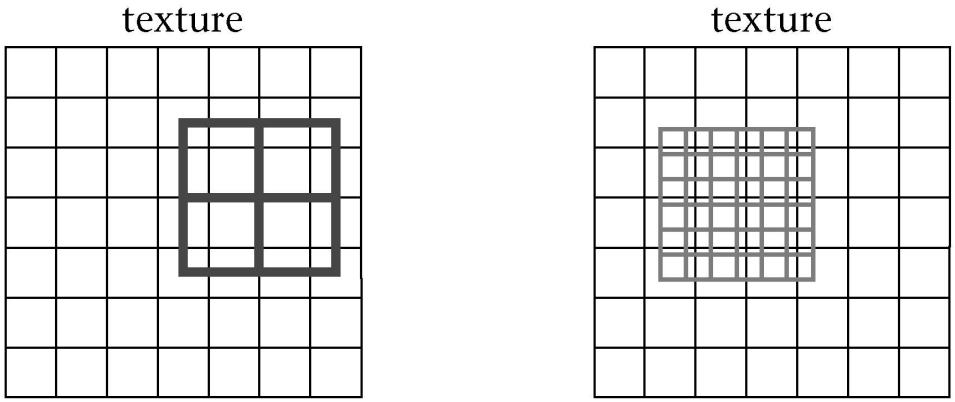


Aliasing

Si ha aliasing ogni qualvolta non ci sia corrispondenza uno-a-uno tra elementi del display (pixel) ed elementi della texture (texel). Distinguiamo due casi:

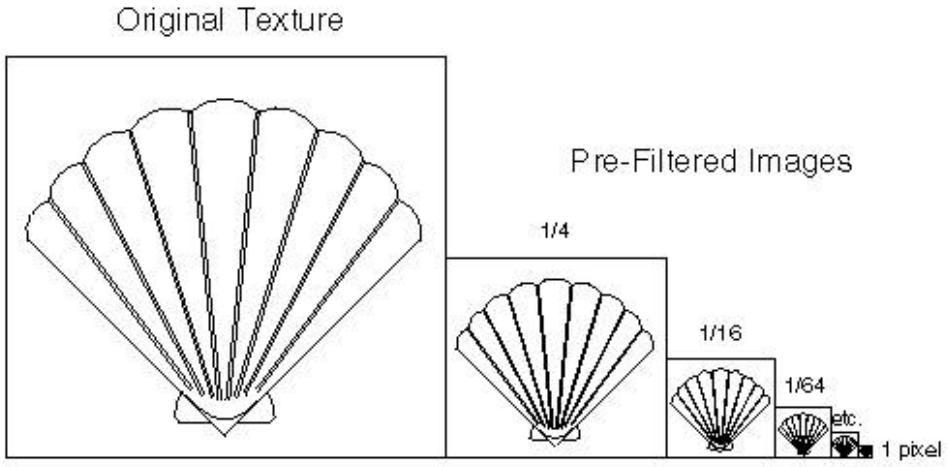
(a) **Magnification**: un texel corrisponde a molti pixel. In questo caso un singolo texel appare come un blocco di pixel nell'immagine. Quello che si vede è una versione ingrandita "a blocchi" della texture. È come ingrandire una immagine digitale: se i pixel diventano troppo grossi si notano effetti "a blocchi".

(b) **Minification**: (il peggiore) un pixel corrisponde a molti texel. Si dovrebbe vedere una versione rimpicciolita della texture. Ma se si usa il metodo delineato prima, in cui ad ogni pixel contribuisce un solo texel, vuol dire che ci sono texel che restano fuori.È come quando si riduce un'immagine digitale: se si sottocampiona si possono perdere particolari.



(a) Minification (b) Magnification

- La soluzione è:
  - per fattori di riduzione moderati, usare interpolazione bilineare nelle coordinate texture per determinare il valore (invece di arrotondare all'intero)
  - per fattori di riduzione significativi (maggiori di due) è necessario tenere conto del valore di tutti i texel che ricadono nella pre-immagine del pixel (che è – grossomodo – più grande di quattro texel), facendone la media.
- L'operazione è costosa da effettuare on-line, per cui si preferisce immagazzinare una versione multi-risoluzione (piramide) della texture, che prende il nome di **mipmap**.



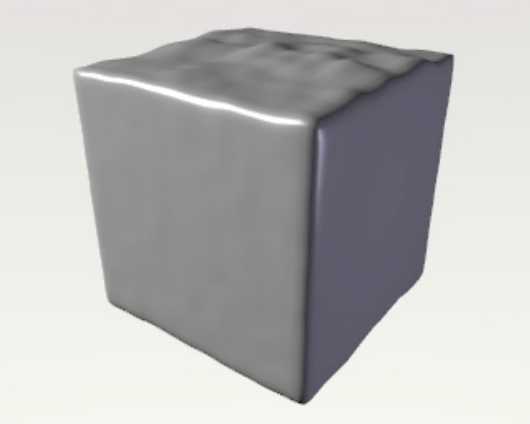
© Woo et al.

- Al momento del rendering, dato un pixel da colorare, si sceglie la texture nella mipmap al livello di dettaglio che minimizza la minificazione (area della pre-immagine del pixel circa 1), e si usa interpolazione bilineare.

Color mapping

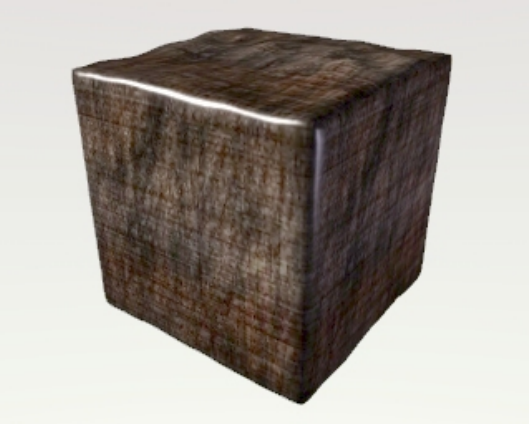
- Una texture di colore è una immagine digitale.
- Nel determinare il colore di un punto il colore dell'elemento della texture corrispondente interagisce con quello assegnato alla superficie dal modello di Phong.
- Questa tecnica si chiama anche più propriamente **color mapping**.
- Il color mapping più semplice (modo **decalcomania**) consiste semplicemente nell'assegnare ad un pixel il colore specificato dalla texture, sovrascrivendo ogni altra informazione (non serve fare shading, verrebbe sovrascritto).
- Un altra tecnica è quella di **modulare** il colore della texture moltiplicandolo per il valore risultante dallo shading.
- In questo caso è tipico assegnare colore bianco (o grigio) alle componenti ambientale, diffusa e speculare modello di Phong della superficie. In questo modo, si ottiene di modulare il colore della texture con l'intensità di illuminaizone della superficie calcolata dal modello di Phong.
- Il problema è che si perdono gli highlights, che assumono il colore della texture. La soluzione consiste nel modulare con la texture separatamente solo le componenti diffusa, ambientale ed emissiva e sommare separatamente alla fine la componente speculare.

Esempio di color mapping



(1) Phong shading

(2) Immagine



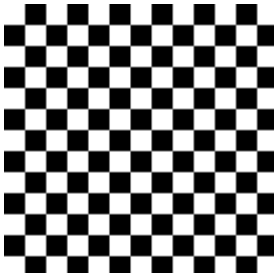
(3) Risultato

Scena con texture sul pavimento, porta e tavolo.

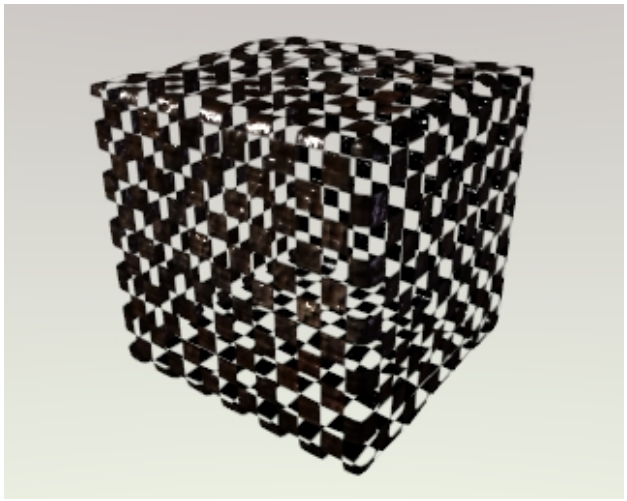
© Alan Watt

Altri usi

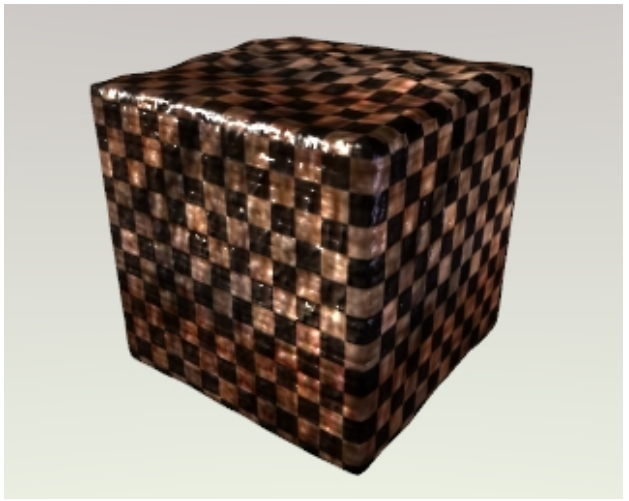
- Le texture vengono solitamente usate per il color mapping.
- Vi sono però altre tecniche che si possono ricondurre all'uso di textures.
  - **Mappa di riflessione:** detta anche **environment map** usa una texture per dare l'impressione che l'oggetto rifletta l'ambiente circostante.
  - **Bump map:** questa tecnica perturba la normale in un punto con il valore corrispondente nella texture; siccome il modello locale di illuminazione usa la normale per calcolare le intensità di colore, il risultato di un bump mapping e di alterare lo shading della superficie (senza modificare ovviamente la geometria)
    Mappa delle normali: estensione della precedente, specifica la normale in ogni punto.
  - **Mappa della luce:** o **light map** serve a contenere il risultato del calcolo della illuminazione compiuto off-line (con modello solitamente globale, view-independent).
  - **Mappa di trasparenza:** si usa per modulare l'opacità dell'oggetto; in tal modo alcune parti possono essere rese trasparenti, altre opache. Spesso (OpenGL) tale informazione viene accorpata ad una texture di colore; la texture conterrà quindi il colore RGB ed un valore di opacità A
  - In generale tutte le caratteristiche che compaiono nel modello locale di illuminazione possono venire modulate dalla texture (mappa di emissione, mappa di specularità, etc...)



(4) Texture



(5) Mappa di trasparenza

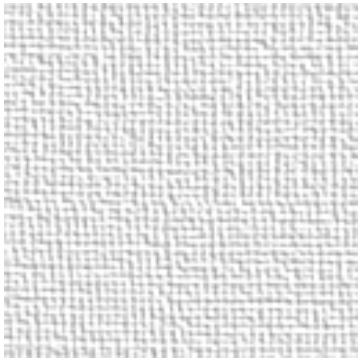


(6) Mappa di emissione

Bump mapping

- Il bump mapping viene usato per fare apparire corrugata (con rigonfiamenti e indentazioni) una superficie liscia (es. arancia).
- Cambiare la geometria per aggiungere i particolari di piccola scala non è in genere fattibile perché richiederebbe un numero troppo elevato di poligoni.
- Invece il bump mapping modifica le normali alla superficie (senza toccare la superficie stessa), così quando viene applicato il modello di illuminazione (Phong) l'apparenze della superficie (il suo "shading") risulta quella della superficie corrugata che si voleva ottenere.

(7) Liscia



(8) Bump map

(9) Risultato

- Sia $P(u, v)$ un generico punto della superficie (parametrizzata) da perturbare.
- Sia data la mappa $B(u, v)$ che specifica la perturbazione (virtuale) da applicare alla superficie, spostando il punto $P(u, v)$ lungo la sua normale della quantità $B(u, v)$. (assumiamo per semplicità $s = u$ e $t = v$).
- La normale in P è dunque (si suppone normalizzata)

$$\mathbf{n} = P_u \times P_v$$

  dove P_u e P_v sono le derivate parziali rispetto ai due parametri
- Se spostassimo P lungo $\mathbf{n}$ di un valore $B(u, v)$ si otterrebbe

$$P'(u, v) = P(u, v) + B(u, v)\mathbf{n}$$

- Per calcolare la nuova normale $\mathbf{n}'$ devo derivare $P'(u, v)$ rispetto a u e v:

$$P'_u = P_u + B_u\mathbf{n} + B\frac{\partial \mathbf{n}}{\partial u} \qquad P'_v = P_v + B_v\mathbf{n} + B\frac{\partial \mathbf{n}}{\partial v}.$$

- Supponendo che $B(u, v)$ sia sufficientemente piccola e la superficie sufficientemente regolare da poter trascurare l'ultimo termine si ottiene

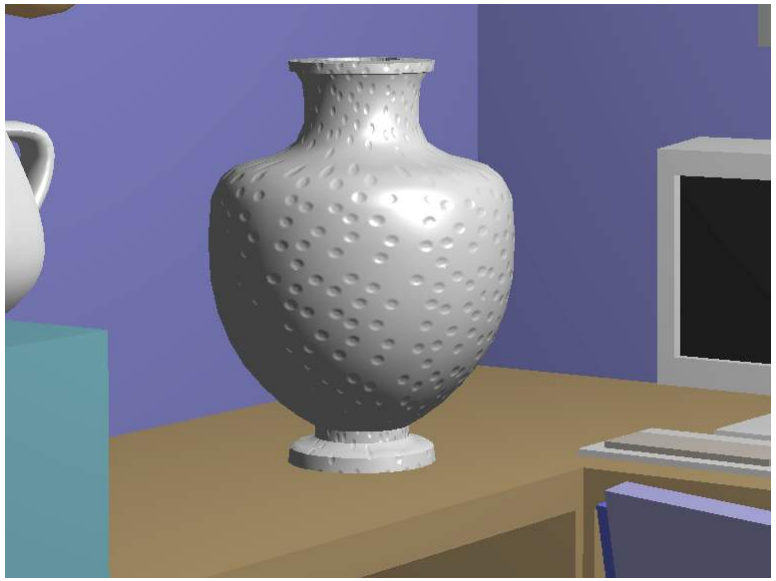
$$\mathbf{n}' = P'_u \times P'_v = \mathbf{n} + B_u\mathbf{n} \times P_v - B_v\mathbf{n} \times P_u$$

- Se applicando il modello di illuminazione a P si usa $\mathbf{n}'$ al posto di $\mathbf{n}$ si ottiene l'impressione

che il punto sia stato perturbato.

- Si noti che il valore $B(u, v)$ non viene usato se non per calcolarne le derivate parziali. Si può memorizzare queste ultime nella bump map, risparmiando tempo di calcolo on-line.
- Alcuni programmi (BMRT) possono perturbare geometricamente il punto P, non solo la sua normale; in tal caso si parla di **displacement map**. È molto più oneroso computazionalmente, anche se i risultati sono realistici se visti da vicino, cosa non vera per il bump-mapping
- Difatti quest'ultimo, modificando solo le normali della superficie, non cambia le proprietà geometriche di questa; se è liscia prima del bump-mapping, lo rimane anche dopo (si vede guardando la silhouette).
- La differenza tra bump-mapping e displacement-mapping è comunque piccola per la maggior parte delle situazioni (telecamera lontana dall'oggetto, superficie non vista di taglio).
- Il bump mapping non è supportato da OpenGL perché la texture viene applicata solo *dopo* il calcolo della illuminazione.

(10) Bump map



(11) Bump mapped object