FONDAMENTI DI COMPUTER GRAPHICS M

LAB 07

Il settimo laboratorio ha riguardato gestione di texture mapping, normal mapping ed environment mapping con OpenGL e GLSL. È stata fornita una scena con alcuni oggetti (toro, colonna, pietra, sfera, cubo, finestre, …) da cui partire. Il laboratorio può essere diviso in quattro parti principali:

1. Applicazioni di semplici texture grafiche, sia a partire da immagini, sia generate proceduralmente;
2. Applicazione del normal mapping;
3. Environment mapping, ovvero la riproduzione di texture ambientali sugli oggetti riflettenti o "trasparenti" (refraction);
4. Gestione delle trasparenze.

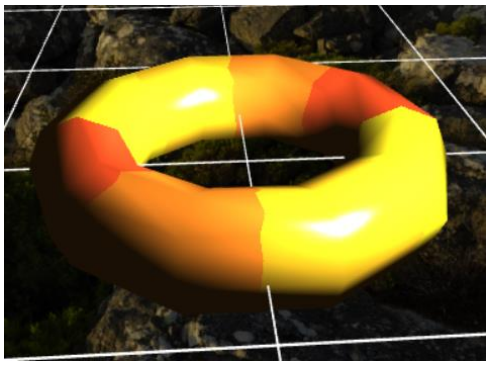
1. Texture mapping, shading e texture procedurali

Il primo step è stato quello di capire come usare OpenGL (e GLSL) per applicare texture, ovvero immagini, agli oggetti presenti in scena. Per farlo, è necessario prima di tutto caricare l'immagine voluta in un array che contiene le informazioni utili, come i canali RGB. Dopo aver applicato alcuni parametri alla texture (come per esempio filtri, tramite la funzione `glTexParameteri`), sarà necessario caricare la texture nella memoria della GPU tramite `glTexImage2D`. A questo punto, basterà associare la texture caricata all'oggetto (tramite id – la texture viene caricata una sola volta, ma può essere utilizzata più volte).

Nel caso sopra descritto, il vertex shader si limita a prendere in input le varie informazioni (matrici) per le trasformazioni e le coordinate texture, passando queste ultime al fragment shader, che, avendo deciso per un texture mapping molto semplice, si limiterà a sostituire il colore del fragment con quello della coordinata texture.

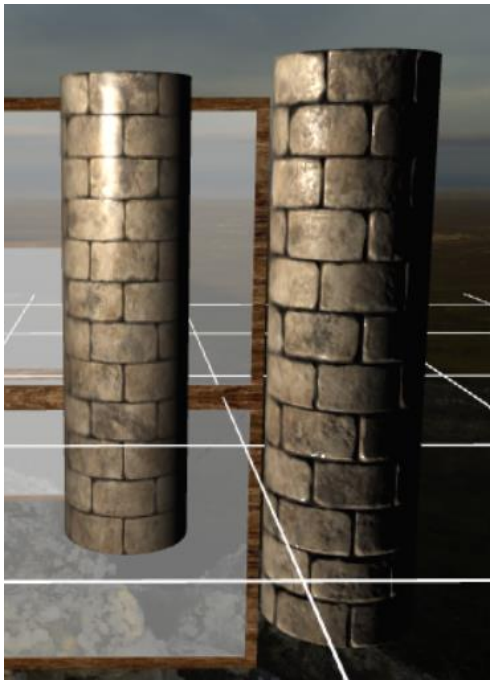
A questo punto, è possibile aggiungere l'illuminazione così come visto nei laboratori precedenti attraverso gli shader (specificando per ogni oggetto, quindi, anche il tipo di materiale, che imposterà il comportamento dell'oggetto rispetto alla luce). Si applica Phong Shading: nel vertex shader si calcolaranno i tre vettori E, N e L, oltre a passare le coordinate texture, mentre nel fragment la texture sarà sostituita alla componente diffusiva (k_d) nella formula per il shading di tipo phong.

Infine, è stato scritto un piccolo algoritmo per la generazione di una texture procedurale (a strisce colorate di 4 colori diversi). Tale algoritmo riempie una matrice di dimensione prefissata e con tre canali di colore (RGB) che andrà a sostituire quella in cui veniva caricata l'immagine di texture precedentemente.



2. Normal mapping

In questa parte di laboratorio, l'obiettivo è stato quello di applicare due texture allo stesso oggetto: una è la classica texture che fornisce il colore del fragment (utilizzata anche in precedenza), mentre la seconda è una *normal map*, ovvero un'immagine in scala RGB che fornisce un valore vettoriale per la modifica della normale alla superficie. Combinando correttamente i valori forniti dalle due texture nella formula per l'illuminazione (Phong), l'effetto ottenuto sarà quello di una superficie che rispetta la geometria della texture caricata (per esempio protuberanze). In realtà, l'effetto è solo visivo, poiché la superificie non viene realmente modificata.

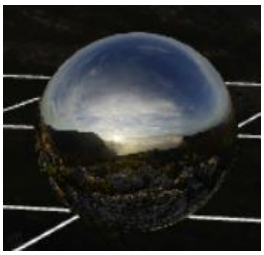


Nell'equazione per l'illuminazione, abbiamo deciso di portare i vettori già presenti nello spazio tangente delle normali: i vettori forniti dalla normal map, dunque, non vengono modificati, mentre dobbiamo operare un cambio di sistema per il vettore `L` ed `E` (luce e riflessione). A questo punto, si utilizzerà il nuovo valore della normale (fornito dalla normal map) per `N` e il valore fornito dalla texture per il `material.diffuse`. Nell'immagine a fianco, alla colonna di sinistra è applicata solo la texture per il colore (e illuminazione), mentre in quella di destra il normal mapping appena descritto.

3. Environment mapping

L'environment mapping serve per applicare texture ambientali e far sì che queste siano riflesse o rifratte dagli oggetti in scena. È necessario un oggetto intermedio, l'*environment map*, che racchiuda tutta la scena e sulla cui superficie viene mappata la texture ambientale. Le texture ambientali sono particolari texture pensate per la superficie di questo oggetto intermedio che nel laboratorio, è un cubo (si avranno quindi sei texture, una per faccia).

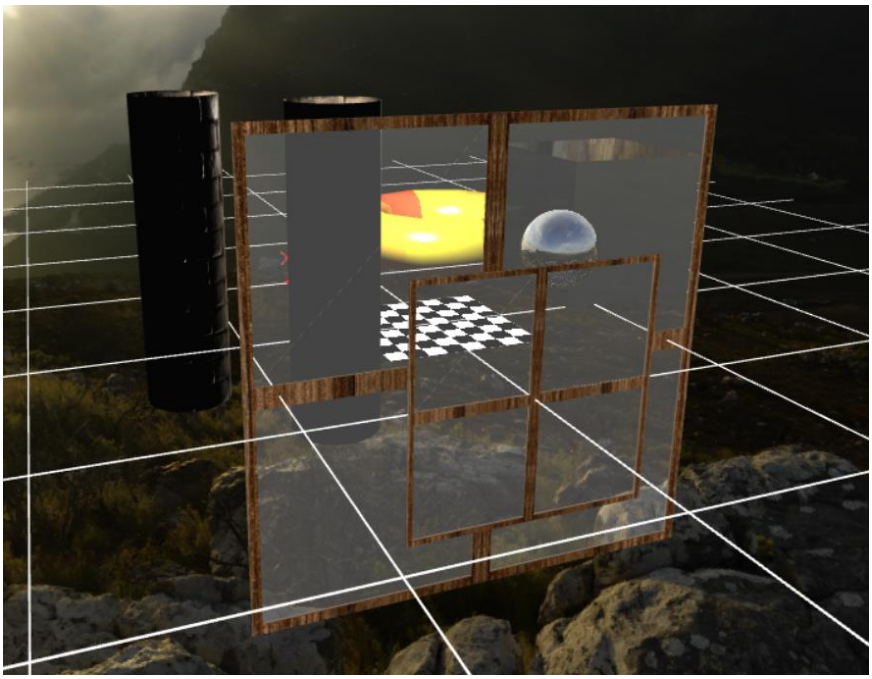
Dopodiché, gli oggetti possono "interagire" con l'ambiente in due modi diversi, tramite riflessione o rifrazione:



- Nel caso della riflessione, il vettore di vista colpirà l'oggetto in scena generando un raggio di riflessione (calcolato da E ed N). Questo raggio di riflessione colpirà l'oggetto intermedio in un punto, punto di cui verrà fatto il sampling rispetto alla texture mappata;
- Nel caso della rifrazione, invece, la direzione del raggio di vista viene leggermente modificata dal cambiamento del mezzo di trasmissione (per esempio da aria a vetro). Il raggio modificato andrà, come prima, a colpire l'oggetto intermedio fornendo il valore per quel punto.

Nell'immagine, da sinistra a destra, un cubo di vetro, di diamante e di ghiaccio.

4. Gestione delle trasparenze



Per la resa semi-trasparente delle due finestre presenti in scena, si è sfruttato il *blending* di OpenGL: una volta calcolato il valore del fragment, questo non viene semplicemente scritto nel frame buffer, ma viene combinato con il valore già presente in esso per quel fragment (se vi sono già degli oggetti in scena, questo corrisponderà al valore del loro fragment).

Nell'esempio, i due valori vengono combinati con una funzione additiva e i parametri *s* e *d*, che indicano la quantità del valore della sorgente (*s* – nuovo fragment) e della destinazione (*d* – vecchio fragment), dipenderanno dal canale alpha della texture della finestra: nei punti in cui

questo sia *<1* il colore finale risultante sarà influenzato anche da ciò che si trova dietro alla finestra.

Per ottenere questo effetto, sarà necessario disegnare gli oggetti di scena per primi e poi le finestre per ultime poiché, altrimenti, il blending non considererebbe tutti i contributi degli oggetti.