Fondamenti di Computer Graphics M
Report LAB 07
SHADER: Texture mapping, Normal mapping, Environmental & Refraction

Giulio Posati

June 1, 2023

Indice

1 Camera motion

L'animazione del movimento della camera in scena è stato realizzato tramite l'inserimento di keyframe: è possibile, infatti, muovere la camera in scena per inserire un nuovo keyframe alla pressione del tasto 'k' e, quando ci sono almeno tre keyframes questi vengono usati come punti di controllo di una curva di Bezier. In particolare vengono calcolati a partire da questi punti di controllo i punti della curva tramite l'algoritmo di deCasteljau (in modo simile a quanto visto per il LAB_01). Premendo la barra spaziatrice si può far iniziare l'animazione che prevede lo spostamento della camera lungo questa curva: l'unica accortezza è stata far coincidere il punto finale della curva con quello iniziale in modo da permettere un loop.

2 Texture mapping 2D

Sono stati realizzati vertex e fragment shader in grado di colorare un oggetto in base ad un immagine, associando le relative coordinate texture ad ogni vertice. Il setup di questa tipologia di shading segue quanto già presente nell'applicazione (lo stesso vale per gli altri punti di questa esercitazione simili a questo).

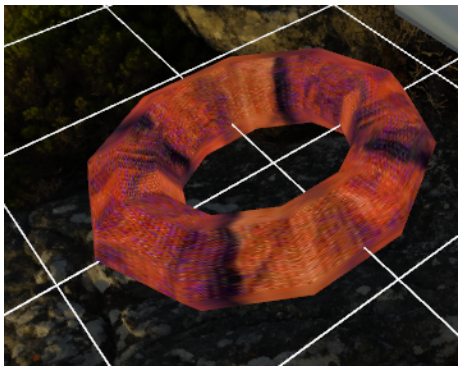


Figure 1: Texture

3 Texture mapping 2D + Shading

In questo caso sono stati realizzati gli shaders che combinano quanto fatto nel punto precedente con il modello d'illuminazione di Phong.

Figure 2: Texture con modello d'illuminazione

4 Procedural mapping

Per questa operazione occorre generare in modo procedurale una texture da applicare all'oggetto toro. Si è scelto di usare una noise texture generata con l'algoritmo di Perlin, il cui codice è stato preso da https://github.com/solprog/Perlin_Noise.

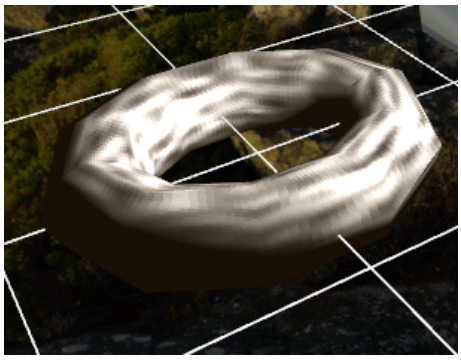


Figure 3: Procedural noise texture

5 Normal mapping

Sono stati realizzati vertex e fragment shader in grado di combinare l'uso di texture, il modello d'illuminazione di Phong e una normal map rgb, in cui le componenti di colore per ogni coordinata texture codificano il vettore normale, simulando un effetto di profondità.

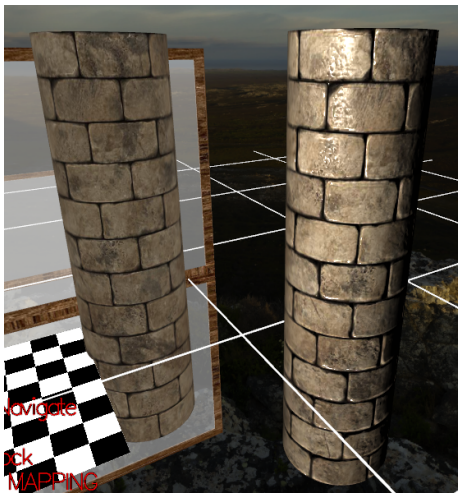


Figure 4: Normal map

6 Environment cube mapping: skybox

Qui occorreva realizzare gli shaders che consentono la creazione di un ambientazione, mappando delle texture su un cubo: oltre a questo è stata rimossa la trasformazione di traslazione al cubo stesso per posizionarlo nell'origine.

Figure 5: Skybox

7 Environment mapping: object REFLECTION

Anche in questo caso è stato realizzato un enveiroment mapping su una sfera in modo che possa riflettere l'ambiente circostante, tramite due shaders appositi.

Figure 6: Riflessione

8 Environment mapping: object REFRACTION

Stavolta è stato realizzato un enveiroment mapping su un cubo in modo che possa simulare l'effetto di rifrazione relativamente all'ambiente circostante, tramite due shaders appositi.

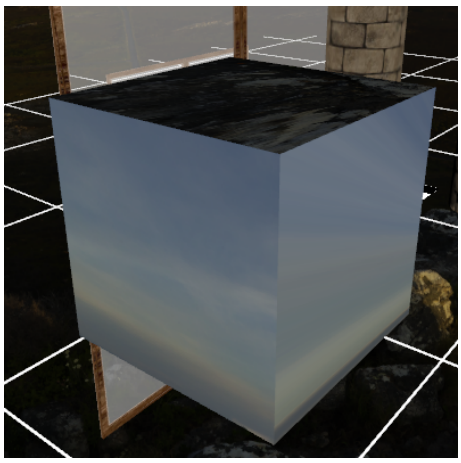
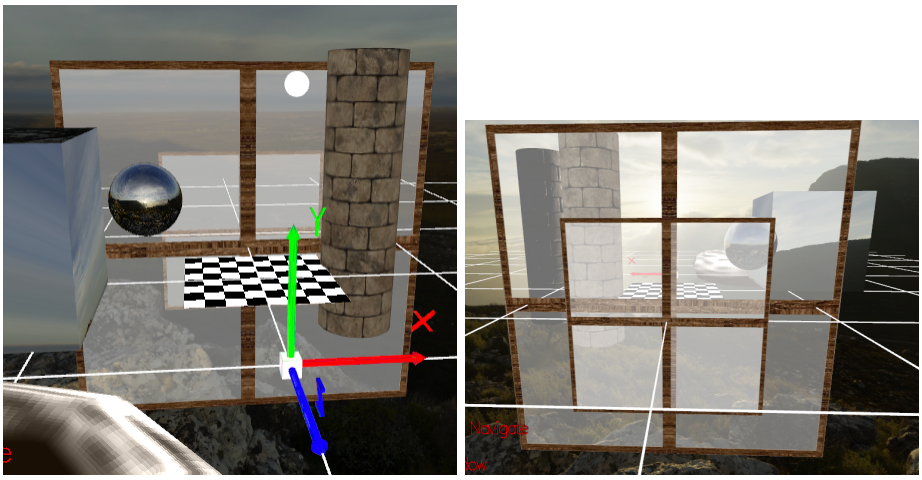


Figure 7: Rifrazione

9 Oggetti semi-trasparenti

Quest'ultima parte prevedeva di andare a modificare la funzione che va a disegnare gli oggetti in scena. La strategia utilizzata consiste nel disegnare prima gli oggetti opachi, disabilitare lo z-buffer ed infine disegnare gli oggetti semi-trasparenti dal più lontano al più vicino alla camera, per poi riattivare lo z-buffer.



(a) Camera davanti agli oggetti (b) Camera dietro agli oggetti

Figure 8: Resa di oggetti semi-trasparenti