

FONDAMENTI DI COMPUTER GRAPHICS LM

LAB 7 - SHADER PART II: NORMAL MAPPING, ENVIRONMENTAL & REFRACTION

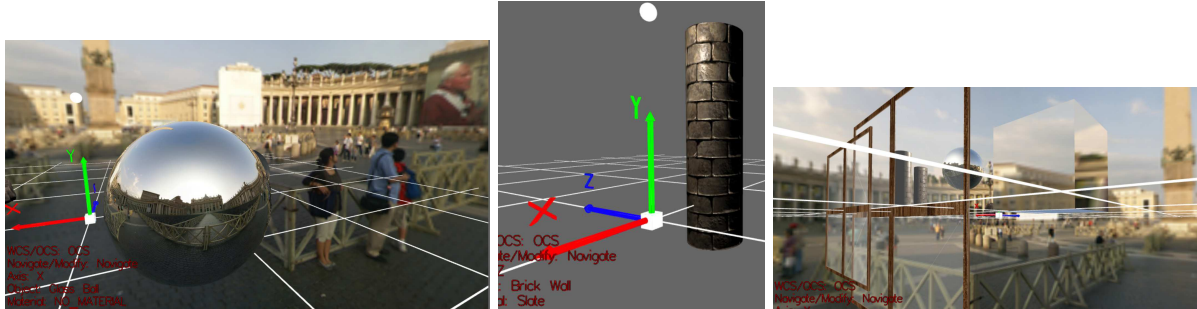


Figure 1: (sinistra) Effetto environment mapping sulla superficie sfera; (centro) normal mapping; (destra) effetto rifrazione attraverso il cubo, effetto trasparenza delle due finestre.

Scaricare i file necessari dalla pagina WEB del docente. L'archivio contiene un semplice programma per la gestione di texture mapping, normal mapping e environment mapping con OpenGL e GLSL, compilare ed eseguire il programma fornito. Il programma è basato sul codice dell'esercitazione 3 ma la resa è basata su shaders. Se lo si ritiene opportuno e utile si possono ereditare dalla esercitazione 3, se svolta, i tool di navigazione in scena e gestione delle trasformazioni, questo upgrade sarà valutato come opzionale.

I tasti frecce dx e sx permettono di selezionare uno tra i seguenti modelli:

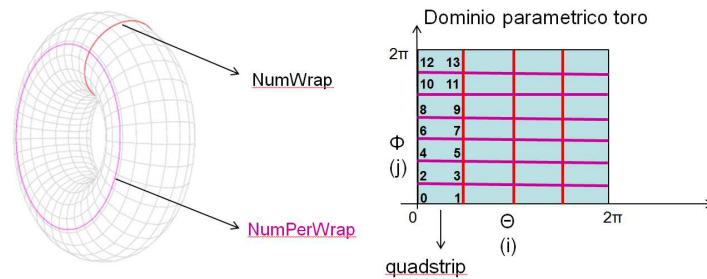
- mesh cubo con 2D texture image caricata da file e associata alle facce.
- mesh toro creato attraverso la tassellazione di una superficie parametrica su un dominio parametrico. Con i tasti *W/w* e *N/n* si incrementa/decrementa il numero di avvolgimenti *W/w* e il numero di vertici per ogni avvolgimento *N/n* del raggio maggiore e del raggio minore.
- mesh sfera smooth (file .obj) creata con normali ai vertici e parametrizzazione associata ai vertici.
- mesh cilindro smooth (file .obj) con 2D texture image caricata da file.

Estendere il programma inserendo la gestione, mediante opportuni vertex e fragment shader, delle seguenti funzionalità:

1. **Normal mapping** di un oggetto mesh (column.obj o sharprockfree.obj) con normal map e immagine texture lette da file, mediante gli shaders *v_normal_map.glsl* (fornito) e *f_normal_map.glsl* (da creare sulla falsa riga di *f_phong.glsl* dell'esercitazione 6) utilizzando multitexturing *GL_TEXTURE0* per DiffuseMap e *GL_TEXTURE1* per NormalMap. L'effetto è illustrato in Fig. 1 centro. Utilizzando lo shader *TEX_PHONG* dell'esercitazione 6 con la DiffuseMap come texture si può notare la differenza tra normal mapping e semplice texture mapping.
2. **Environment cube mapping: skybox** Gli shaders *v_skybox.glsl* e *f_skybox.glsl* realizzano l'effetto di resa dell'ambiente circostante mappato su un cubo che contiene la scena. Gestire il posizionamento e dimensionamento del cubo nell'applicazione affinché sia visibile l'effetto env map dell'ambiente di background come illustrato in Fig. 1 sinistra.

3. **Environment mapping: object REFLECTION** Realizzare gli shaders *v_reflection.glsl* e *f_reflection.glsl* per combinare l'effetto di resa dell'ambiente circostante su un oggetto riflettente in scena (es. sfera). Utilizzare il cube mapping come superficie intermedia; l'effetto sulla sfera é illustrato in Fig. 1 sinistra.
4. **Environment mapping: object REFRACTION** modificare il punto precedente per gestire un oggetto trasparente in scena (es. cubo). Realizzare gli shaders *v_refraction.glsl* e *f_refraction.glsl*.
5. **OPZIONALE: Oggetti semi-trasparenti: gestire in scena un paio di oggetti semi-trasparenti** (es. *window.obj*) facendone la resa in *display()* dal piú lontano al piú vicino dopo aver reso tutti gli oggetti opachi.

Osservazione: Tassellazione e parametrizzazione della superficie toro Si ricorda che il toro ha la seguente rappresentazione parametrica $\mathbf{S}(\theta, \phi)$ con $\theta, \phi \in [0; 2\pi]$:



$$\begin{aligned}
 x(\theta, \phi) &= \sin(\theta)(R + r \cos(\phi)) \\
 y(\theta, \phi) &= \cos(\theta)(R + r \cos(\phi)) \\
 z(\theta, \phi) &= r \sin(\phi)
 \end{aligned}$$

Il vettore normale è definito come

$$\mathbf{n}(\theta, \phi) = \mathbf{S}_\theta(\theta, \phi) \times \mathbf{S}_\phi(\theta, \phi)$$

con derivate parziali

$$\begin{aligned}
 \mathbf{S}_\theta(\theta, \phi) &= \begin{bmatrix} dx/d\theta \\ dy/d\theta \\ dz/d\theta \end{bmatrix} = \begin{bmatrix} \cos(\theta)(R + r \cos(\phi)) \\ -\sin(\theta)(R + r \cos(\phi)) \\ 0 \end{bmatrix} \\
 \mathbf{S}_\phi(\theta, \phi) &= \begin{bmatrix} dx/d\phi \\ dy/d\phi \\ dz/d\phi \end{bmatrix} = \begin{bmatrix} -r \sin(\theta) \sin(\phi) \\ r \cos(\theta) \sin(\phi) \\ r \cos(\phi) \end{bmatrix}
 \end{aligned}$$