# Grafica

*Anno Accademico 2018-2019*

# Esercitazione 7

Matteo Berti, Matricola 889889

27 giugno 2019

**Comandi da tastiera**

I comandi di navigazione dell'esercitazione 3 sono presenti anche qui.

**Note**: il codice è stato generato e testato utilizzando le seguenti specifiche:
- OS: Debian GNU/Linux 9 (stretch)
- IDE: CLion 2018.3.4

All'interno della funzione `init()` viene settata la working directory corrente, se si utilizzano parametri diversi da quelli indicati è possibile sia necessario riadattarla.

**UPGRADE: integrazione tool di navigazione**
I tool di navigazione integrati sono: zoom, pan orizzontale e verticale, traslazione/rotazione/ridimensionamento per ogni oggetto in scena sia in *WCS* che in *OCS*.
Extra: la traslazione e rotazione della luce è gestita opportunamente come nell'esercitazione 6.

**1. Normal mapping**
Per il normal mapping, nel fragment shader, viene calcolata la normale attraverso la `NormalMap`, ovvero si prende tramite la funzione `texture()` il valore di un texel, mappato nell'intervallo $[0, 1]$ e lo si trasla nell'intervallo $[-1, 1]$, tipico delle normali. Con questa normale è calcolato normalmente *Phong* e alla termine il colore del frammento tiene in considerazione sia il texel della `DiffuseMap` che le componenti luminose calcolate.

**2. Environment cube mapping: skybox**
La skybox è stata posizionata al centro della scena e scalata di 500 volte per "inglobare" tutti gli oggetti presenti in scena, camera compresa.

**3. Environment mapping: object reflection**
Il fragment shader *reflection* non fa altro che calcolare il vettore riflesso rispetto al vettore che dal frammento punta alla camera, tramite la funzione `reflect()`. Restituisce infine il colore del texel della skybox colpito dal vettore ottenuto.

**4. Environment mapping: object refraction**
Il fragment shader *refraction* è del tutto simile al reflection con la differenza che viene utilizzata la funzione `refract()`, a cui è passato anche un indice di rifrattività, in questa esercitazione è usato quello del vetro: $\frac{1.00}{1.52}$.

**OPZIONALE: oggetti semi-trasparenti**
All'interno della funzione `display()` viene calcolato quale dei due oggetti semi-trasparenti è più

lontano alla camera, questo verrà disegnato per primo.

Il risultato finale è il seguente: