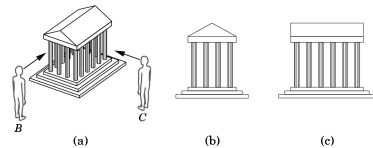


Sintesi di Immagini

- ❖ Metafora fondamentale

Object vs viewer

- ❖ Object (*scene*): rappresentazione digitale (forma e caratteristiche) di un oggetto reale tridimensionale
- ❖ Viewer: *strumento* che permette di ottenere da un object un immagine
- ❖ *Rendering* è il processo con cui un viewer genera un immagine a partire da una scena.

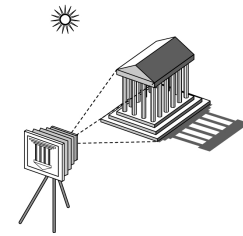


1

Caveat

- ❖ Object e viewer, come tutte le metafore, sono entità non definite rigidamente...

- ❖ La luce fa parte del viewer?
- ❖ Il viewer è anch'esso un object?

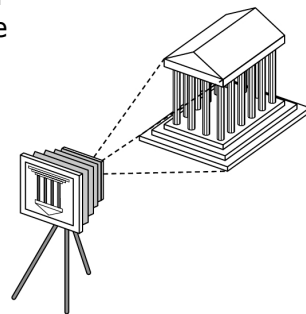


2

Sintesi di Immagini

Tra le caratteristiche parametrizzabili di un viewer la più evidente è la *Camera*:

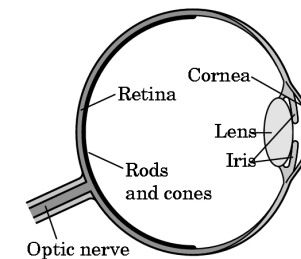
- ❖ L'insieme di quei parametri che definiscono come e dove si guarda una certa scena.



3

Rendering: Approccio Fisico

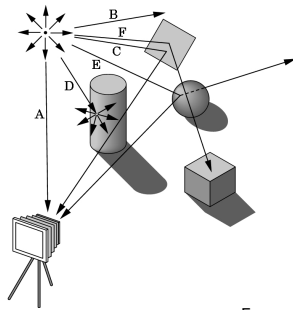
- ❖ Come si svolge fisicamente il processo della visione?



4

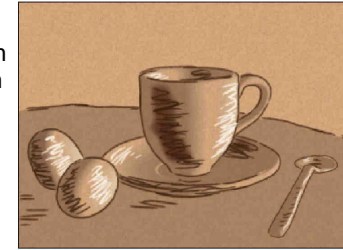
Simulare l'illuminazione

- ❖ Fotorealismo
- ❖ La simulazione il più dettagliata possibile di tutte le interazioni tra la luce e gli oggetti.



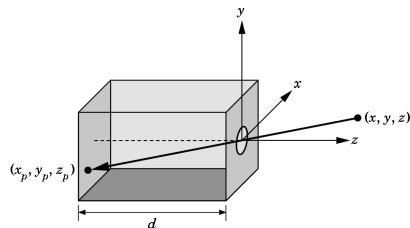
Rendering Approccio Non fisico

- ❖ *NPR (non photorealistic rendering)*
- ❖ Simulare il processo con cui un artista genera un'immagine
- ❖ Settore piuttosto nuovo e di ricerca



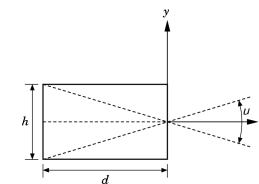
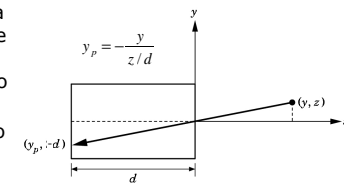
Pin hole Camera

- ❖ Il processo con cui si formano le immagini può essere simulato da una scatola chiusa con un foro infinitesimamente piccolo sul davanti
- ❖ minima macchina fotografica



Pin hole Camera

- ❖ In una pinhole camera è facile determinare come si forma l'immagine sul fondo della camera (piano della pellicola)
- ❖ Il pinhole è detto il centro di proiezione



Pin Hole camera

- ❖ La pinhole camera e' un modello astratto
 - ❖ Fuoco infinito
 - ❖ Luminosità infinitesima
- ❖ In realtà (cioè nelle macchine fotografiche e nell'occhio) si sostituisce il pin hole con una lente
 - ❖ Profondità di campo limitata
 - ❖ Maggior luminosità
 - ❖ Distorsioni varie

9

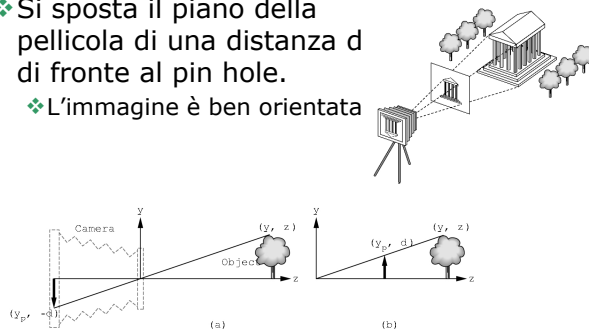
Pin Hole Camera

- ❖ Nelle prossime lezioni assumeremo sempre che stiamo utilizzando una pin hole camera.
- ❖ Cio' non toglie che si possano usare modelli più sofisticati che simulino tutte le altre caratteristiche delle camere reali (occhio e macchine fotografiche)

10

Modello standard della PIC

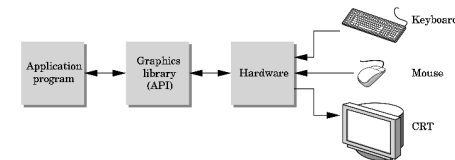
- ❖ Si sposta il piano della pellicola di una distanza d di fronte al pin hole.
 - ❖ L'immagine è ben orientata



11

Definire una camera?

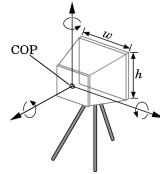
- ❖ Definire i parametri di una camera è necessario perché un viewer possa generare un'immagine di una scena
- ❖ Interattivamente (implicitamente) a (CAD, Games)
- ❖ Seguendo una API (esplicitamente)
 - ❖ E.g. using an interface between a program and a graphic system
 - ❖ OpenGL, DirectX Java3d etc



12

Definire una camera

- ❖ Di solito si deve specificare
 - ❖ Posizione (del centro di proiezione)
 - ❖ Orientamento
 - ❖ Lunghezza focale: determina la grandezza sul piano immagine



13

Definire una Camera

OpenGL

- ❖ `gluLookAt(center_of_projection, look_at_point, up_direction)`
- +
- ❖ `glPerspective(Field_of_view, ...)`

14

Frame buffer

- ❖ Una porzione di memoria dedicata alla memorizzazione dell'immagine come insieme di pixel da mostrare a video.
- ❖ Caratteristiche
 - ❖ Risoluzione (numero di pixel)
 - ❖ Range tipici 320x200 <- > 1600x1200
 - ❖ Profondità (bit per pixel)
 - ❖ Range tipici 1 <-> 32 (128)

15

Architettura di un renderer

- ❖ La pipeline di rendering; assumendo che
 - ❖ La scena è composta di entità geometriche semplici (primitive) descritte per mezzo di vertici
 - ❖ L'algoritmo di rendering che voglio usare è strutturato in maniera da processare e disegnare tutte le primitive una alla volta abbastanza indipendentemente (object order)
- ❖ Allora per ogni primitiva le operazioni da fare sono, in sequenza, le seguenti



16

Pipeline di rendering

- ❖ Il fatto di strutturare il rendering
 - ❖ Indipendentemente per primitiva
 - ❖ Per ogni primitiva in una pipeline ben determinata
- ❖ Permette di progettare hw grafico che espliciti il parallelismo nei due livelli
 - ❖ Multiple rendering pipelines
 - ❖ I passi più lenti della pipeline possono essere parallelizzati più massicciamente



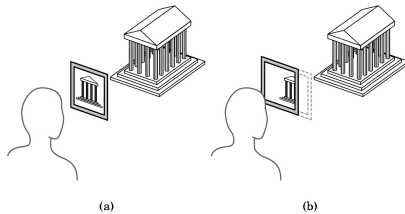
18

Transformazioni di modellazione

- ❖ Ogni oggetto nella scena ha, di solito il proprio sistema di riferimento
- ❖ I vertici della scena da rendere devono essere trasformati in un unico sistema di riferimento: quello della camera.

Lighting e Clipping

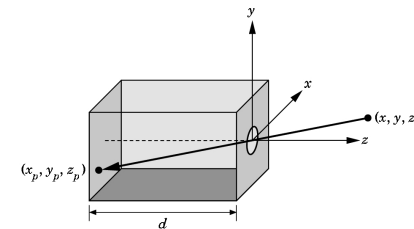
- ❖ Dopo la trasformazione di modellazione si può decidere che cosa è visibile per la camera corrente (e quindi interrompere la pipeline per ciò che non è visibile)



19

Proiezione

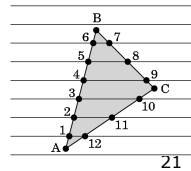
- ❖ Si calcola dove ogni vertice cade nel piano di proiezione



20

Rasterizzazione

- ❖ Per ogni primitiva a questo punto sappiamo dove finiscono nel frame buffer i suoi vertici.
- ❖ Il processo di trovare tutti i pixel che nel frame buffer appartengono alla primitiva è detto rasterizzazione.



Caveat

- ❖ Sulla pipeline di rendering torneremo più volte
- ❖ Gli step possono essere ben più dettagliati
- ❖ Questa pipeline di rendering NON è l'unica esistente
- ❖ Non tutti gli step, non su tutti gli hw, sono implementati effettivamente in hw