H1 Rendering

La definizione di rendering presa in considerazione è la stessa usata qui.

Abbiamo due classi di algoritmi usati per trasformare i modelli 3D in immagini raster, che sono in grado cioè di calcolare il colore che ogni pixel del monitor deve avere:

- *Ray Tracing*
- *Rasterization based*

H2 Ray Tracing vs Rasterization based

I due approcci sono l'uno l'opposto dell'altro, e questo influisce soprattutto sul costo dell'algoritmo:

```
*** RAY TRACING ***
FOREACH pixel p on the screen
    FOREACH primitive o
```

Complessità $\approx O(P \times M)$, dove P è il numero di pixel dello schermo e M è il numero di primitive della scena.

```
*** RASTERIZATION ***
FOREACH primitive o
    FOREACH pixel p of o
```

Complessità $\approx O(M \times N)$, dove M è il numero di primitive della scena e N è il numero di pixel "coperti" dalle primitive della scena.

Poiché vale che $N \leq P$, il costo computazionale dell'algoritmo rasterization based è in genere minore.

Rasterization Based	*Ray Tracing*
✔ **veloce**: complessità lineare con numero di primitive, processo solo le primitive coinvolte ✔ **programmato in HW** ✔ altamente **parallelizzabile**	✔ simile ai **metodi naturali** di acquisizione delle immagini ✔ **facile** renderizzare effetti di luce più complessi ✔ altamente **parallelizzabile**
✘ **difficile** renderizzare effetti di luce più complessi, ma possibile (uso di trucchi SW)	✘ **difficile** eseguirlo in modo veloce, ma possibile (uso di ray tracing in HW)