CREAZIONE DI MAPPE CON

Lezione 6

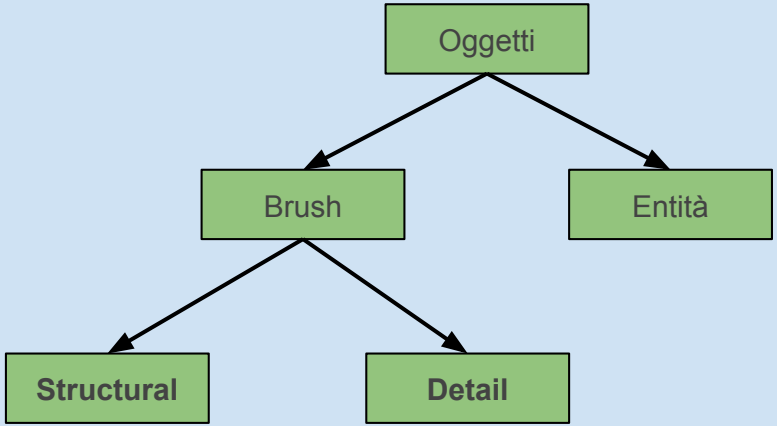
Ottimizzazione

Ottimizzazione

- Ottimizzare significa:
  - Far girare meglio il livello mentre viene giocato (principalmente usare Caulk, comunque UrT gira bene dappertutto)
  - Far compilare la mappa più velocemente

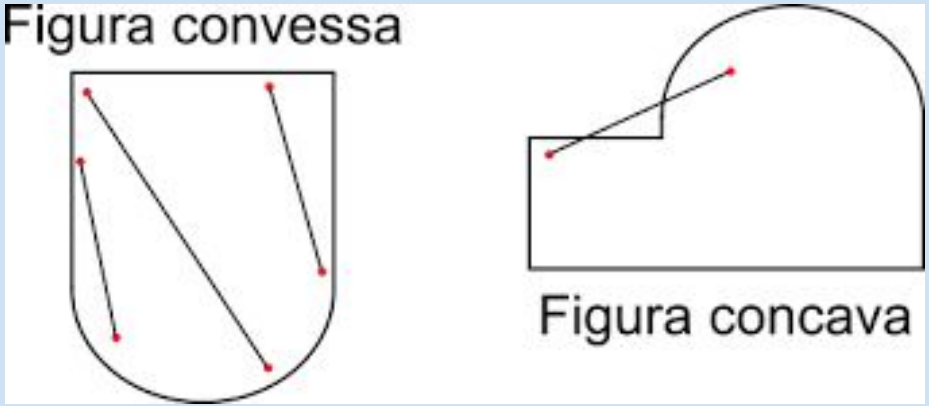
Ottimizzazione

- I brush si dividono in due tipi:
  - **Structural:** generano *vis-leafs* e *portals*
  - **Detail:** non generano *vis-leafs* e *portals*



Definizione utile

- I volumi del vostro livello sono concavi o convessi. Una forma è:
  - **Convessa:** se il segmento che unisce due suoi punti qualsiasi è interamente contenuto nella figura stessa
  - **Concava:** esiste almeno un segmento che congiunge due punti della figura ma non è del tutto contenuto in essa



Compilatore in dettaglio

FASE BSP

(vengono creati portali, *leaf node* e *bsp-tree*)

FASE VIS

(vengono creati la tabella PVS e il file .prt)

Fase BSP (Binary Space Partition)

- Processo ricorsivo (che si ripete uguale finché non finisce) nel quale i volumi vuoti del livello vengono suddivisi finché non rimangono soltanto volumi **convessi** (*leaf nodes*).

- Ogni volume convesso si connette a un altro tramite un portale. Portali e *leaf nodes* sono segnati nel *BSP tree*.

Fase VIS (Visibility)

- Le forme convesse sono più semplici per capire quali parti del livello sono visibili da una data prospettiva.

- In questa fase il compilatore prende ogni portale e controlla quali altri portali sono visibili intorno a lui e segna il risultati nella tabella PVS (*Potentially Visible Set*).

- Crea il file .prt che permette di visionare i *leaf nodes*.

A cosa serve tutto questo?

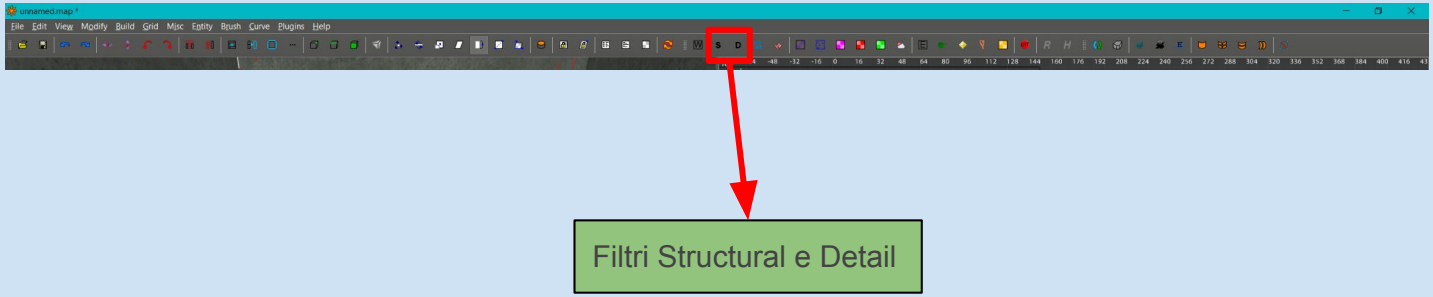
- Queste informazioni vengono usate nel gioco per non calcolare le parti del livello non visibili dal giocatore, risparmiando così potenza di calcolo.

- Controllare quali portali sono visibili è un processo che può richiedere molto tempo al compilatore.

- Ci sono alcuni Brush che non devono poter generare *leaf nodes*, e questi devono essere resi **Details.**

Come convertire a Detail

- Tutti i Brush sono Structural di default.
- Premendo **Ctrl+D** si filtrano i Details (cioè si vedono solo gli structural).
- Premendo **Alt+D** si converte a Detail.
- Per ottimizzare bisogna lasciare Structural solo le pareti principali della nostra mappa, cioè quelle che bloccano effettivamente la vista.



File .prt

- Si può visionare il file .prt all'interno di NetRadiant per scovare gli ultimi Brush da rendere Detail.

- Dopo aver compilato, andate su NetRadiant e sotto *plugins* → *prtview* scegliete la voce *Load .prt file.*

- Potete terminare di vederlo premendo *Unload .prt file.*

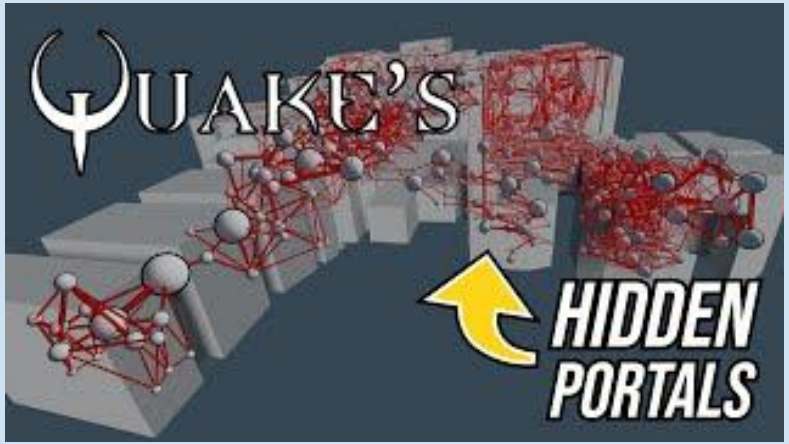
Caulk-Hull Methodology

- Il Caulk è intangibile e invisibile, ma genera *leaf nodes*.

- Un altro modo per ottimizzare è il **Caulk-Hull**: Rendere tutto il livello Detail, e creare uno “scheletro” fatto di Caulk per guidare la fase VIS.

- Consiglio di provarlo in progetti futuri, comunque alla fine del processo di creazione.

Video utili per capire BSP e VIS (in inglese)



Luci

- Id Tech 3 non supporta luci o riflessi dinamici. Meglio mettere meno luci ma più intense che viceversa perché altrimenti aumenta il tempo di compilazione.

- Le luci sono in realtà delle texture statiche applicate in fase di compilazione che fanno sembrare illuminato un dato brush, chiamate **lightmap**. Vengono applicate in base ai valori di intensità, distanza e colore delle luci vicine al brush.

  Vengono creati anche dei **light volumes** intorno alle luci, per illuminare le entità mobili.

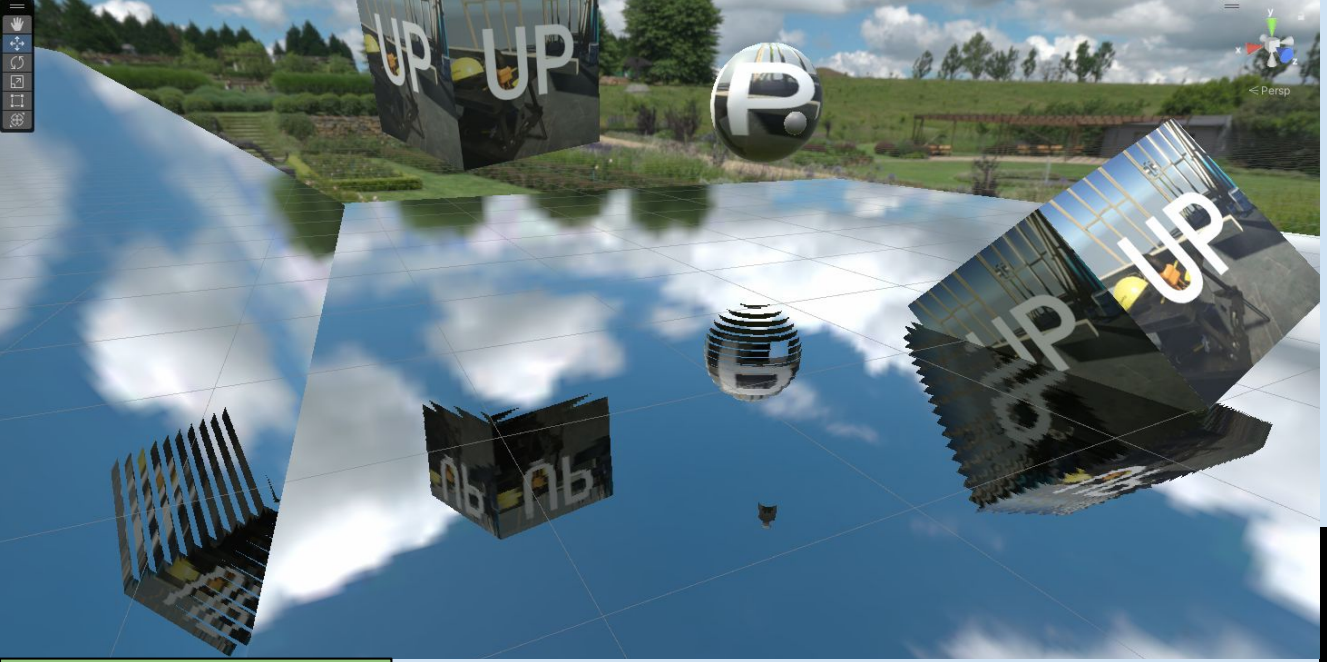
Video utile per capire le luci (in inglese)

Uno sguardo al futuro

- Gli engine più moderni supportano luci dinamiche e anche calcoli dei riflessi, attraverso due metodi principali:

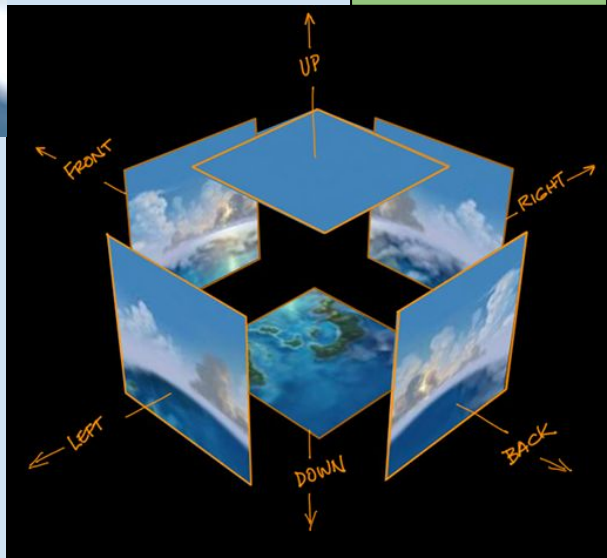
  - **Cubemaps:** Viene fatta una “foto” nelle 6 direzioni intorno al cubemap (entità piazzata manualmente), queste vengono applicate alle texture circostanti per simulare dei riflessi. Completamente statici, possono deformarsi.

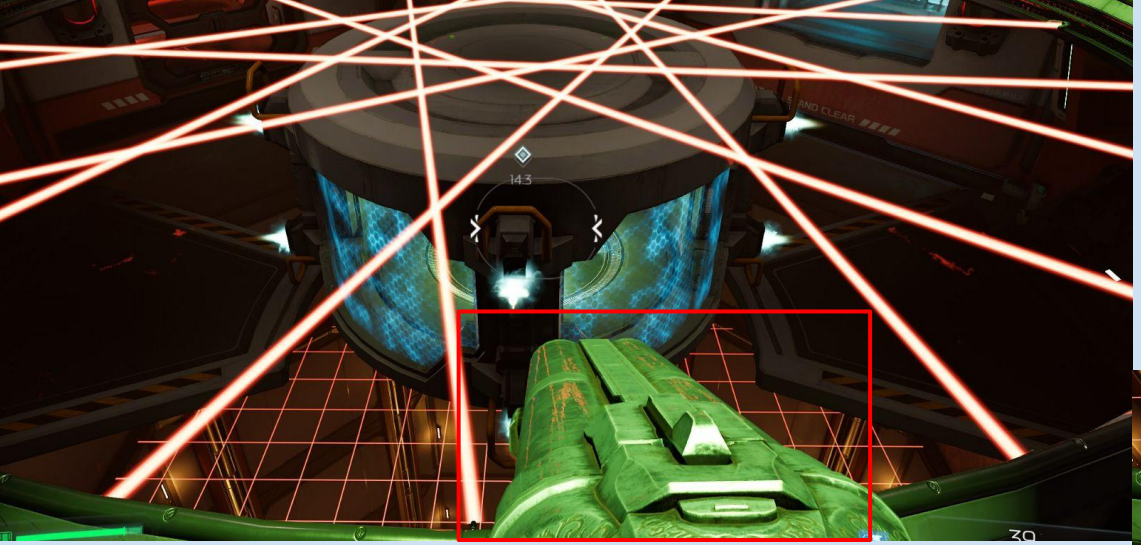
  - **Screen Space Reflections (SSR):** Come suggerisce il nome, viene riflesso solo ciò che è presente sullo schermo. In tempo reale, ma gli oggetti non inquadrati non vengono riflessi.

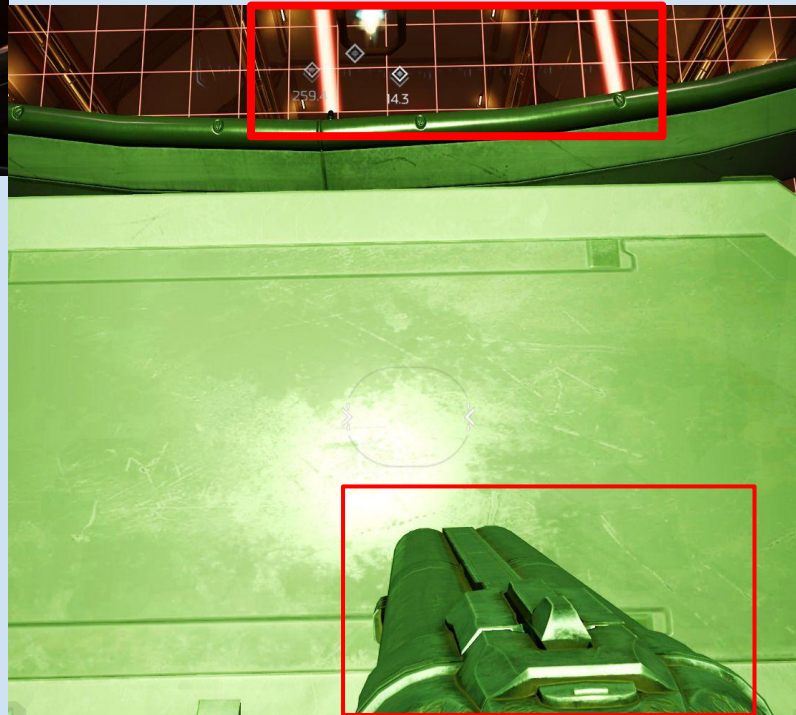


Screen Space Reflection

Cubemap







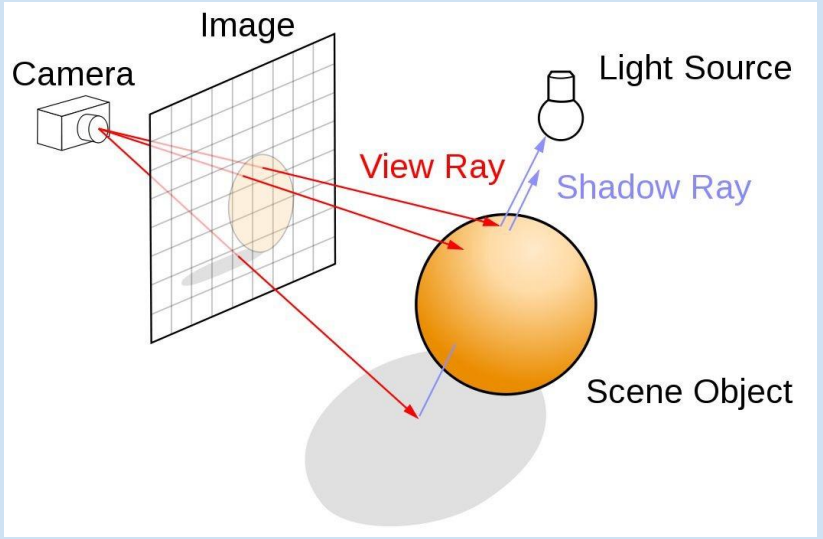
SSR in Doom (2016) [id Tech 6]:

Nell’immagine di sopra i laser vengono riflessi sul fucile perché li sto guardando, in quella sotto non vengono riflessi anche se dovrebbero esserlo.

Uno sguardo al futuro

- Id Tech 7 e 8 supportano il **raytracing**, tecnica che simula il percorso reale dei raggi di luce, in particolare il modo in cui rimbalzano, vengono assorbiti, si riflettono o si rifrangono sulle superfici degli oggetti in una scena 3D.

  Questo permette di avere luci, ombre e riflessi altamente realistici, ad un costo molto elevato di performance però.



Raytracing in Doom Eternal (2020) [id Tech 7]:

il fulmine viene riflesso sul vetro anche se non è visto dal giocatore

Raytracing nella vita reale (facoltà di medicina a Terni): i fotoni rimbalzano sui vetri molto riflettenti dell’edificio e entrano nell’obiettivo della fotocamera, mostrando in tempo reale le scale e le persone.

Full Compile

- Avendo ottimizzato la nostra mappa, possiamo fare un full compile (final).