Alcune basi matematiche del Ray-Tracing

- ✓ Vediamo, a titolo di esempio:
  - ⇒ come rappresentare un raggio
  - ⇒ Come rappresentare alcune **primitive di rendering** (piani e sfere)

    Ciò che può essere renderizzato direttamente (tutto il resto, deve essere *scomposto* in primitive) Quindi, per gli algoritmi di ray-tracing: ciò che sono in grado di intersecare con un raggio
  - ⇒ come manipolare un raggio (creazione raggio primario, computo raggi secondari di riflessione, rifrazione, shadowing)
  - ⇒ come computare l'interesezione di un raggio con alcune **primitive di rendering** (piani e sfere)

27

Costruzione del raggio primario (per un pixel)

- ✓ Punto di partenza $\mathbf{p}$:
  - ⇒ il punto di fuoco della pin-hole camera
  - ⇒ cioè la posizione del «buco di spillo»
  - ⇒ cioè il Point of View (PoV)
- ✓ Direzione $\vec{\mathbf{d}}$:
  - ⇒ Passo 1: individuare la posizione $\mathbf{q}$ del pixel sul piano immagine
  - ⇒ La direzione è data da

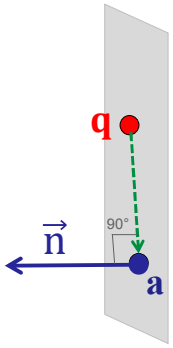
$$\vec{\mathbf{d}} = \frac{\mathbf{q} - \mathbf{p}}{\|\mathbf{q} - \mathbf{p}\|}$$

La normalizzazione non è strettamente necessaria, ma avere raggi con direzione espressa da un vettore unitario può essere comodo

30

Esempio: Intersezione Raggio – Piano 1/3

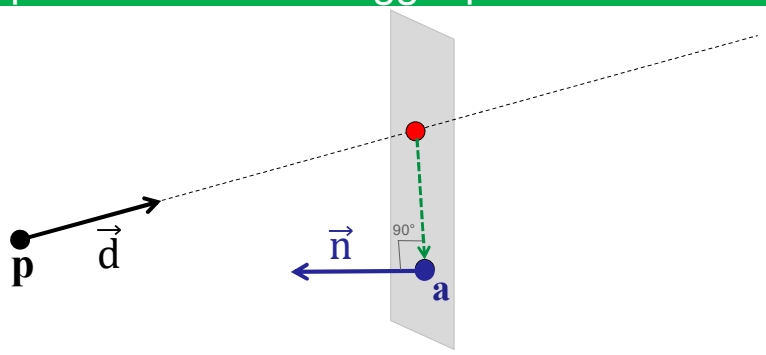
- ✓ piano definito da: un punto su di esso **a** e il suo vettore normale $\vec{n}$ (l'orientamento del piano)
- ✓ punti su piano: cioè tutti i punti **q** tali che il vettore $(\mathbf{a} - \mathbf{q})$ è ortogonale ad $\vec{n}$, cioè $(\mathbf{a} - \mathbf{q}) \cdot \vec{n} = 0$
- ✓ intersecare il piano con un raggio dato $(\mathbf{p}, \vec{d})$ = trovare un k (se ∃) t.c. $\left(\mathbf{p} + k\vec{d}\right)$ sia un punto sul piano



Piano passante per il punto **a** e avente normale $\vec{n}$

31

Esempio: Intersezione raggio piano 2/3



intersecare raggio con sfera = trovare un $k \geq 0$ (se ∃) tale che

Il vettore verde

$$\left(\mathbf{a} - \left(\mathbf{p} + k\vec{\mathrm{d}}\right)\right) \cdot \vec{n} = 0$$

Il punto rosso

perché so che il vettore verde deve essere ortogonale ad $\vec{n}$

32

Esempio: Intersezione raggio piano 3/3

Trovo k (che è l'unica incognita) manipolando l'equazione

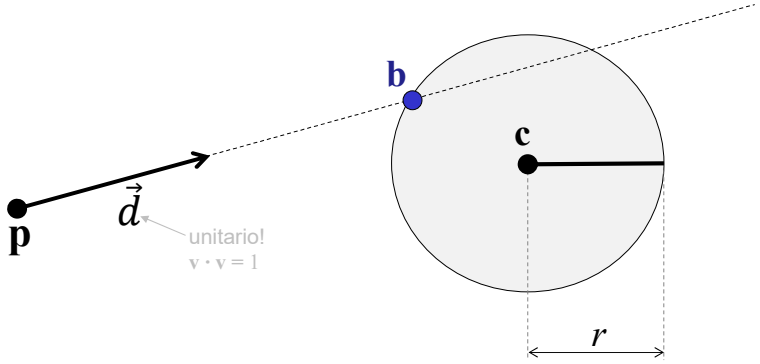
$$\left(\mathbf{a} - \left(\mathbf{p} + k\vec{\mathrm{d}}\right)\right) \cdot \vec{\mathrm{n}} = 0$$
$$\Leftrightarrow$$
$$\left((\mathbf{a} - \mathbf{p}) - k\vec{\mathrm{d}}\right) \cdot \vec{\mathrm{n}} = 0$$
$$\Leftrightarrow$$
$$(\mathbf{a} - \mathbf{p}) \cdot \vec{\mathrm{n}} - k\left(\vec{\mathrm{d}} \cdot \vec{\mathrm{n}}\right) = 0$$
$$\Leftrightarrow$$
$$k = \frac{(\mathbf{a} - \mathbf{p}) \cdot \vec{\mathrm{n}}}{\vec{\mathrm{d}} \cdot \vec{\mathrm{n}}}$$

Se $\vec{d} \cdot \hat{n} = 0$, div per zero.
Ma se è così, allora
il raggio è ortogonale a $\hat{n}$
cioè corre parallelo
al piano: no intersez.

Se $k > 0$, allora il raggio interseca il piano, e l'intersezione è alla posizione $\mathbf{a} + k\vec{\mathrm{d}}$
altrimenti, il raggio non interseca il piano

33

Esempio: Intersezione raggio sfera 1/3



sfera (come superficie implicita):

$$f(\mathbf{q}) = \mathbf{0} \quad \text{con} \quad f(\mathbf{q}) = \|\mathbf{q} - \mathbf{c}\|^2 - r^2$$

intersecare raggio con sfera = trovare un $k \geq 0$ (se ∃) t.c.

$$f\left(\mathbf{p} + k\vec{d}\right) = \mathbf{0}$$

34

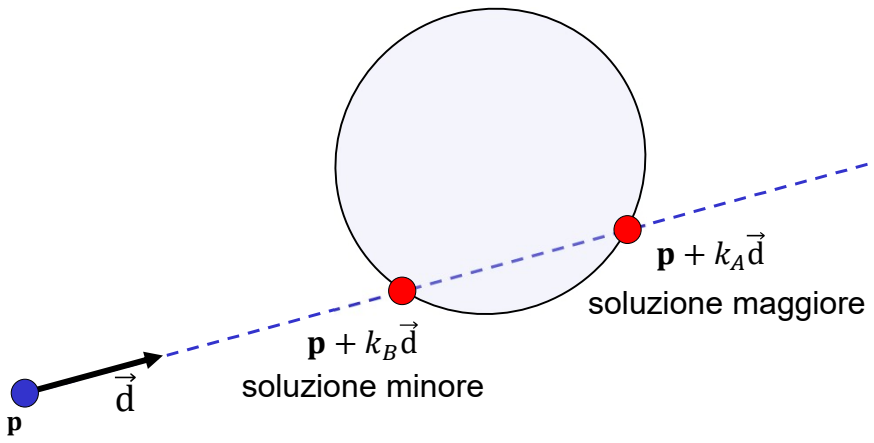
Esempio: Intersezione raggio sfera 2/3

$$f\left(\mathbf{p}+k\vec{d}\right)=0$$
$$\Leftrightarrow$$
$$\left\|\mathbf{p}+k\vec{d}-\mathbf{c}\right\|^{2}-r^{2}=0$$
$$\Leftrightarrow$$
$$\left\|(\mathbf{p}-\mathbf{c})+k\vec{d}\right\|^{2}-r^{2}=0$$
$$\Leftrightarrow \quad \|\vec{v}\|^2=\vec{v}\cdot\vec{v}$$
$$\left((\mathbf{p}-\mathbf{c})+k\vec{d}\right)\cdot\left((\mathbf{p}-\mathbf{c})+k\vec{d}\right)-r^{2}=0$$
$$\Leftrightarrow$$
$$k^{2}\underbrace{\left\|\vec{d}\right\|^{2}}_{a}+k\underbrace{2\,(\mathbf{p}-\mathbf{c})\cdot\vec{d}}_{b}+\underbrace{\|\mathbf{p}-\mathbf{c}\|^{2}-r^{2}}_{c}=0$$

Equazione 2° grado in k . Prendo la soluzione minore delle 2.
Se è >0 allora abbiamo trovarto k e quindi anche il punto di intersezione

35

Esempio: Intersezione raggio sfera 3/3



$$k_{A,B}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

devo ovviamente considerare la soluzione minore
(quella col segno $-$)

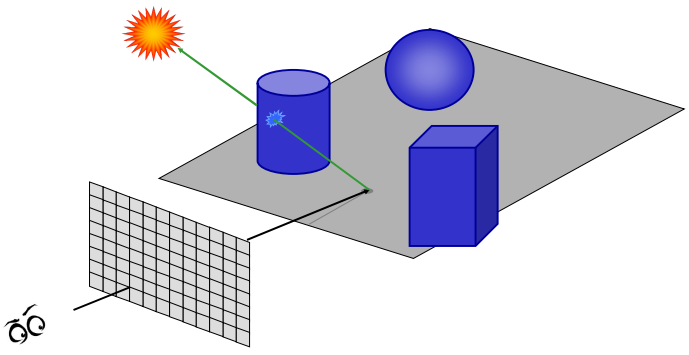
36

Ray Tracing (classe di algoritmi)

- ✓ Seguendo la stessa idea per un altro passo…
- ✓ **Tracciamo** a ritroso il percorso dei fotoni
  - ⇒ non solo dal POV ad un punto su un oggetto 3D,
  - ⇒ ma dall'oggetto 3D all'emettitore della luce che ha emesso quel fotone
- ✓ Questo richiede di computare le intersezioni su un ulteriore raggio
  - ⇒ Detto di shadowing
  - ⇒ E' un esempio di raggio «secondario» (ne vedremo altri)
  - ⇒ Nota: è la *stessa* procedura, «intersezione raggio-primitiva», eseguita più volte per uno stesso pixel.
- ✓ Algoritmi basati su tracciamento di raggi sono detti di Ray-Tracing (compreso il semplice ray-casting)

37

Ray-Tracing: ombre portate (nette)

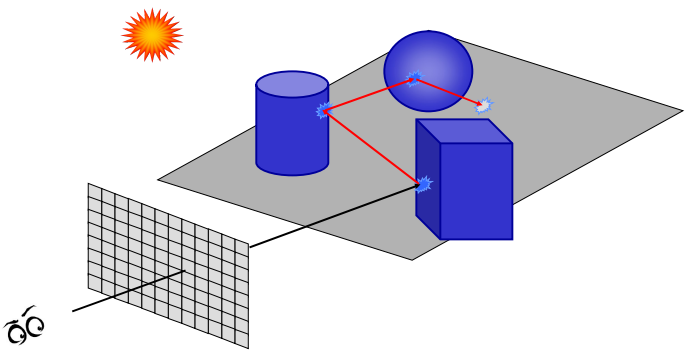


raggio primario

raggio secondario **di shadowing**
(qui: interseca, quindi il punto colpito dal raggio primario è in ombra)

38

Ray-Tracing: riflessioni speculari

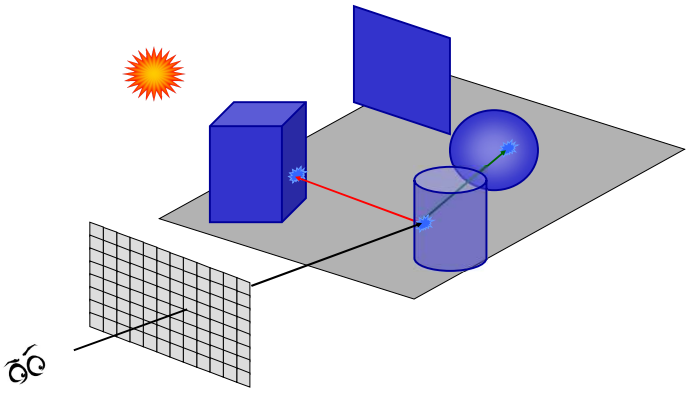


raggio primario

raggi secondari di riflessione

39

Ray-Tracing: semitrasparenze con rifrazioni



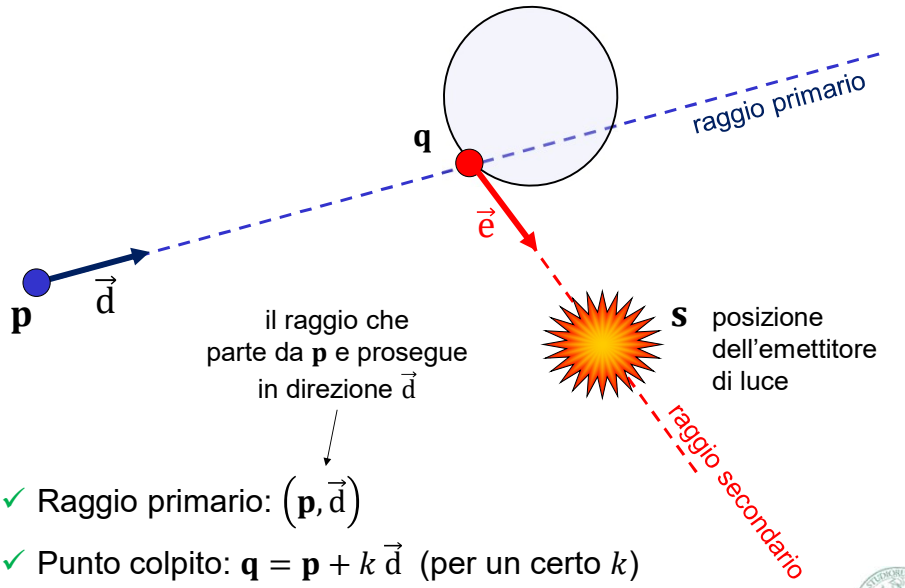
raggio primario

raggio secondario di riflessione

raggio secondario di rifrazione

40

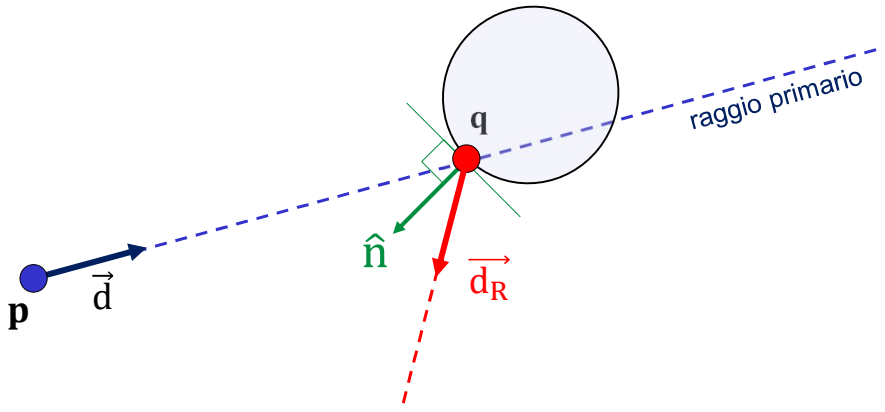
Computo del raggio secondario di shadowing



- ✓ Raggio primario: $(\mathbf{p}, \vec{d})$
- ✓ Punto colpito: $\mathbf{q} = \mathbf{p} + k\,\vec{d}$ (per un certo k)
- ✓ Raggio di shadowing: $(\mathbf{q}, \vec{e})$, con $\vec{e} = \frac{(\mathbf{s}-\mathbf{q})}{\|\mathbf{s}-\mathbf{q}\|}$

41

Computo del raggio secondario di riflessione

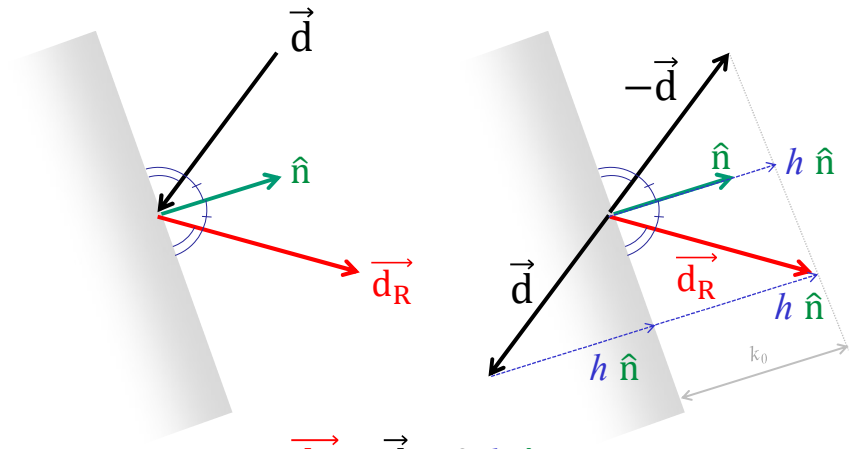


- ✓ Raggio primario: $(\mathbf{p}, \vec{d})$
- ✓ Punto colpito: $\mathbf{q} = \mathbf{p} + k\,\vec{d}$ (per un certo k)
- ✓ Raggio di riflessione: $(\mathbf{q}, \vec{d_R})$,
  con $\vec{d_R}$ il vettore $\vec{d}$ riflesso da un piano di normale $\hat{n}$

Normale della
superficie in **q**

42

Computo di una direzione riflessa da un piano



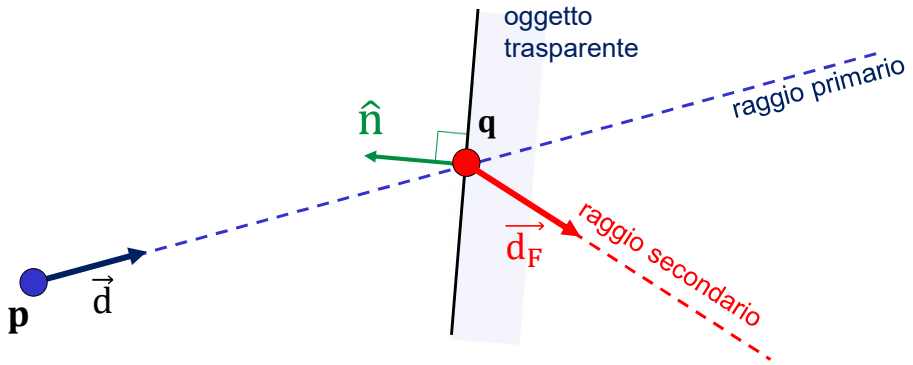
$$\vec{d_R} = \vec{d} + 2\ h\ \hat{n}$$

dove

$$h = (- \vec{d} \cdot \hat{n})$$

43

Computo del raggio secondario di rifrazione



- ✓ Raggio primario: $(\mathbf{p}, \vec{d})$
- ✓ Punto colpito: $\mathbf{q} = \mathbf{p} + k\ \vec{d}$ (per un certo k)
- ✓ Raggio di rifrazione: $(\mathbf{q}, \vec{d_F})$,
  con $\vec{d_F}$ il vettore $\vec{d}$ rifranto da un piano di normale $\hat{n}$

La formula della direzione rifranta $\vec{d_F}$ (che non vediamo) dipende dall'«*indice di rifrazione*» del materiale

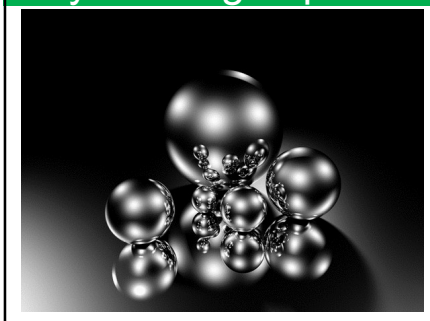
44

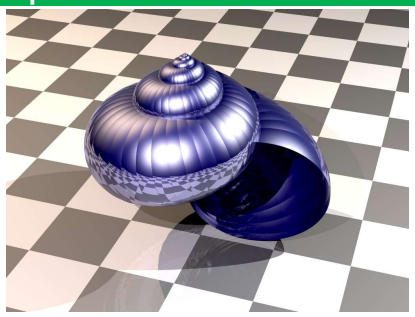
Come computare la direzione raggio riflesso?

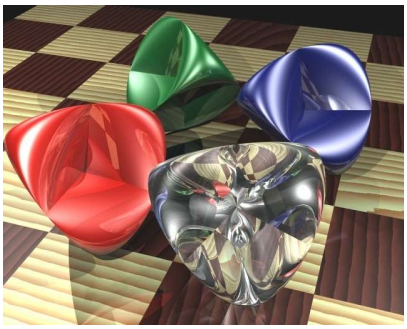
- ✓ Input:
  - ⇒ Direzione del raggio primario
  - ⇒ Normale della superficie nel punto colpito dal raggio primario
- ✓ Output:
  - ⇒ Direzione del raggio secondario
  - ⇒ (insieme alla posizione colpita dal raggio primario, questo modella il raggio secondario di riflessione)

45

Ray-Tracing : tipici esempi di risultati







(Advanced Rendering Toolkit - Alexander Wilkie - 1999)

(with CUDA)

50

Algoritmi di Ray-tracing: efficienza

- ✓ Un'implementazione triviale per
  - ⇒ una scena con M oggetti («primitive»)
  - ⇒ immagini N×N pixel
  - ⇒ con K raggi per pixel (1 primario e K-1 secondari)

  richiede ben N^2 M K intersezioni raggio-primitiva!
  - ⇒ È un numero molto molto elevato
  - ⇒ Complessità temporale dell'algoritmo: $O(N^2 M K)$
  - ⇒ Per questo motivo, questo tipo di algoritmi è usato soprattutto per il **rendering offline**
- ✓ Sono possibili però ottimizzazioni…
  - ⇒ per testare solo un sottoinsieme delle primitive
  - ⇒ **Implementazioni parallelizzate** attraverso GPU (infatti ogni pixel può essere computato indipendentemete dagli altri: elevato livello di **parallelismo implicito**)

51

Ray tracing: alcune varianti

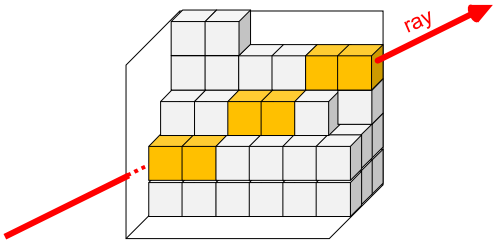
Nota: questa terminologia può variare leggermente da una fonte all'altra

- ✓ Molti algoritmi e varianti si basano su principi simili.
- ✓ Alcuni di questi:
  - ⇒ Ray-casting:
    - un solo raggio (primario) viene "mandato" (*cast*) per pixel
  - ⇒ Ray-tracing:
    - Ogni raggio viene "tracciato" (*traced*) attraverso vari fenomeni successivi riflessione o rifrazione, attraverso raggi secondari
    - Nota: in ordine inverso rispetto alla realtà, come al solito
    - (è anche il usato come nome della classe generale di questi algoritmi)
  - ⇒ Path-tracing:
    - Per ogni pixel, lanciare molti raggi, che seguono i path pseudo-casuali ("monte carlo sampling") con una distribuzione che riflette quella reale
    - Mediare i risultati per ottenere il pixel finale
    - Più raggi vengono mandati, maggiore il realismo
  - ⇒ Ray-marching:
    - Una classe di algoritmi per accelerare l'intersezione raggio-primitiva
    - Principio: raggio viene spezzato in una serie di passi (segmenti) successivi, per ciascuno dei quali si devono testare solo le primitive a lui vicine (invece di tutte)

53

Esempi: Ray tracing su alcuni tipi di modelli visti

- ✓ Raytracing di volume di voxel:
  (con ciascun voxel = valore di densità)
  - ⇒ È una forma commune di **direct volume rendering**
  - ⇒ Per ogni raggio primario, identifico i voxel attraversati
  - ⇒ Assegno il pixel corrispondente ad una funzione di tutti valori di intesità dei voxel attraversati
  - ⇒ per esempio, banalmente, il tono di grigio corrispondente al valore max

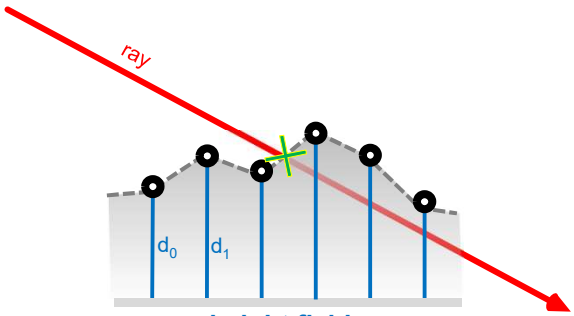


griglia di voxel

54

Esempi: Ray tracing su alcuni tipi di modelli visti

- ✓ Raytracing di Campi di Altezza (schema)
  - ⇒ Tracciare il raggio sulla griglia 2D di altezze x, y
  - ⇒ Trovare la prima posizione in cui la z del raggio cade sotto l'Altezza a coordinate $[x][y]$



height field
(vista laterale)

55

Esempi: Ray tracing su alcuni tipi di modelli visti

✓ Raytracing di CSM (schema):

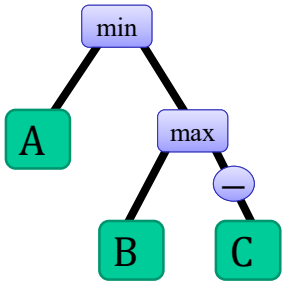
⇒ Trovare *tutte* le intersezioni $\mathbf{p} + k_i\vec{\mathrm{d}}$ con *tutte* le foglie della funz F

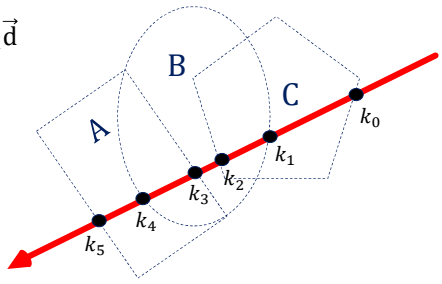
⇒ Ordinarle per k_i

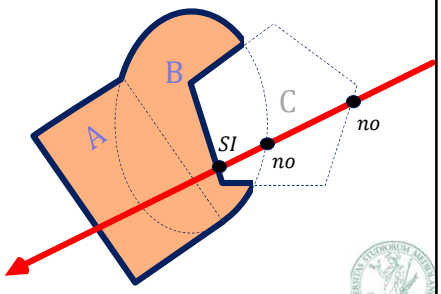
⇒ Trovare la prima intersezione in cui $F\left(\mathbf{p} + k_i\vec{\mathrm{d}}\right) = 0$

⇒ Esempio con:

$$F = \min\left(\mathrm{A}, \max\left(\mathrm{B}, -\mathrm{C}\right)\right)$$







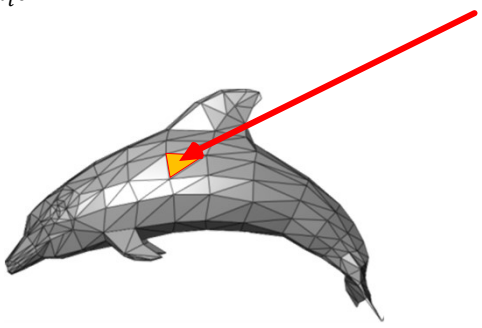
56

Esempi: Ray tracing su alcuni tipi di modelli visti

✓ Raytracing di tri-mesh

⇒ Per ogni triangolo: trovare l'intersezione $\mathbf{p} + k_i\vec{\mathrm{d}}$ raggio-triangolo, se esiste (come implementare questo calcolo?)

⇒ Prendere l'intersezione con il k_i minore



E' una soluzione molto generale, perché è sempre possibile tradurre i nostri modelli 3D in una mesh. Ma può essere una soluzione molto onerosa, (richiede ottimizzazioni, che non viediamo) e la discretizzazione introduce errori

57