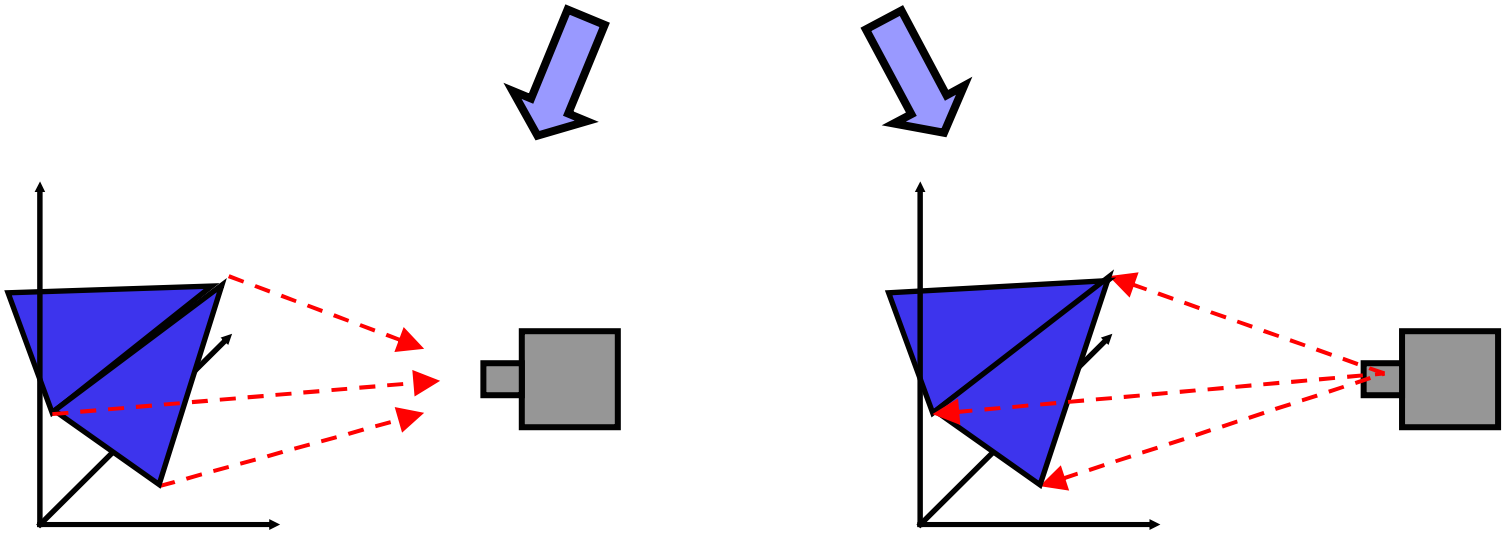
Grafica Computazionale

Ray-Tracing

Fabio Ganovelli
fabio.ganovelli@isti.cnr.it
a.a. 2005-2006

Ray-Tracing e Rasterization

Rendering in Computer Graphics



Rasterization:
Proiettare la scena sullo schermo

Ray Tracing:
Proiettare i pixel sulla scena

Rasterization vs. Ray Tracing

- Rasterizzazione

  Dato un insieme di raggi e di primitive, calcolare il sottoinsieme di raggi e di primitive

  Usa una griglia 2D come struttura di indicizzazione per l'efficienza

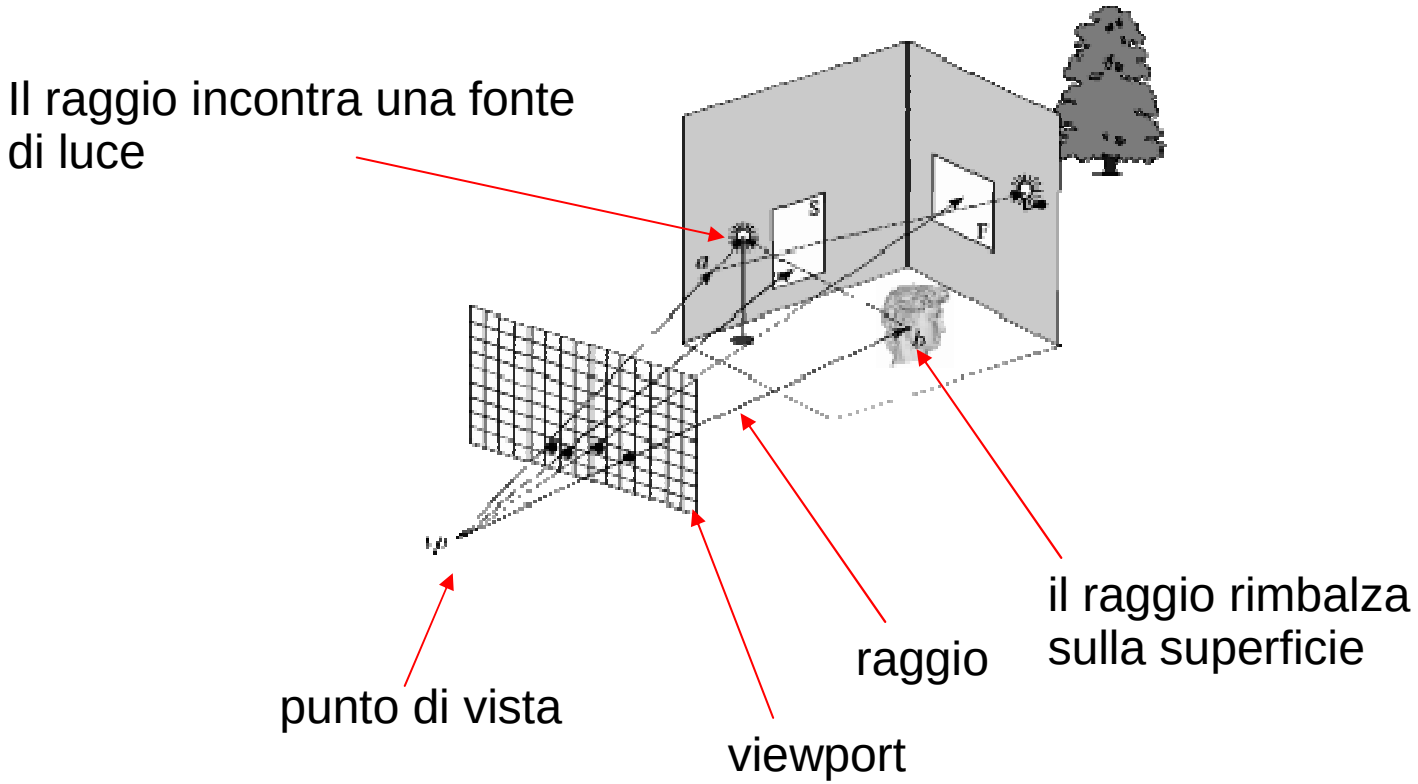
- Ray Tracing

  Dato un insieme di raggi e di primitive, calcolare il sottoinsieme di primitive colpite dai raggi

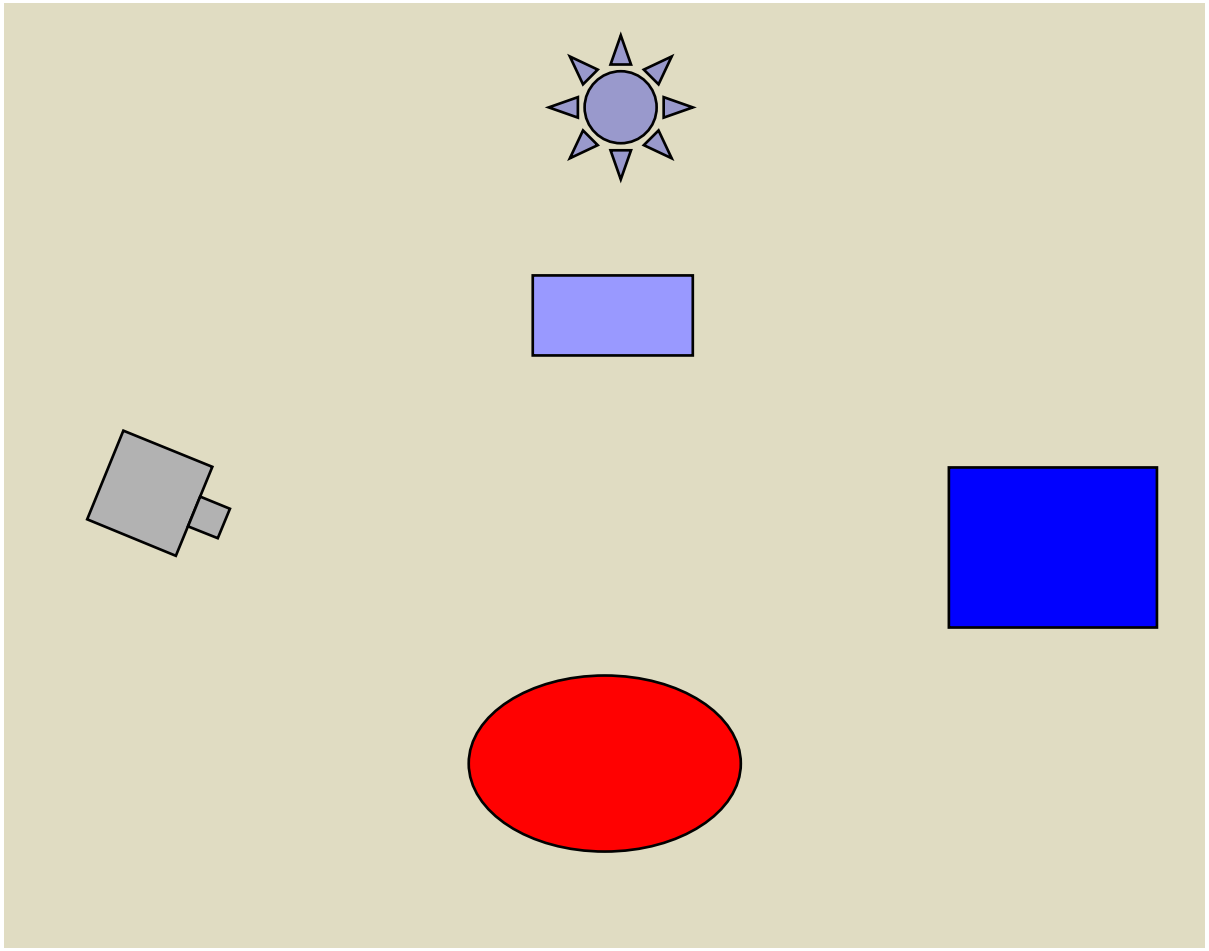
  Usa una struttura gerarchica tridimensionale per efficienza

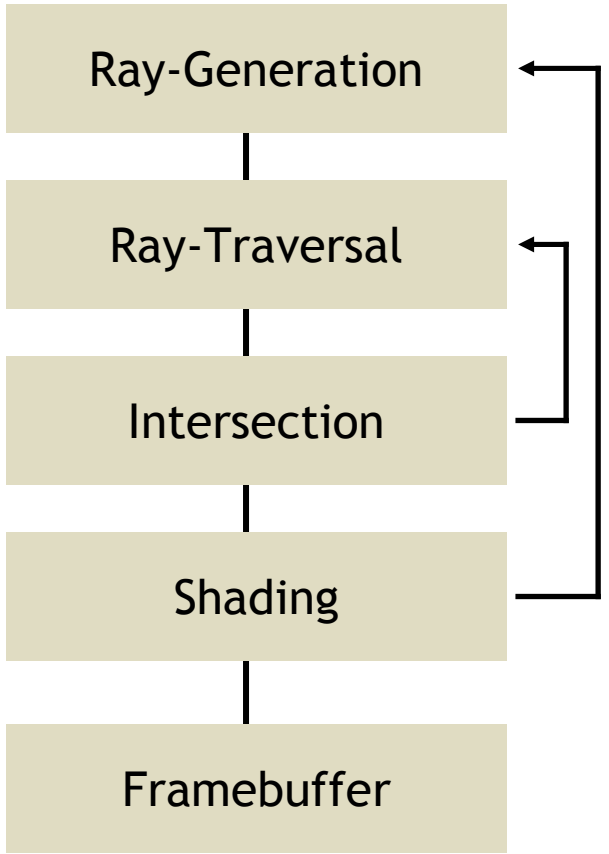
Memento

- Dalla seconda lezione….

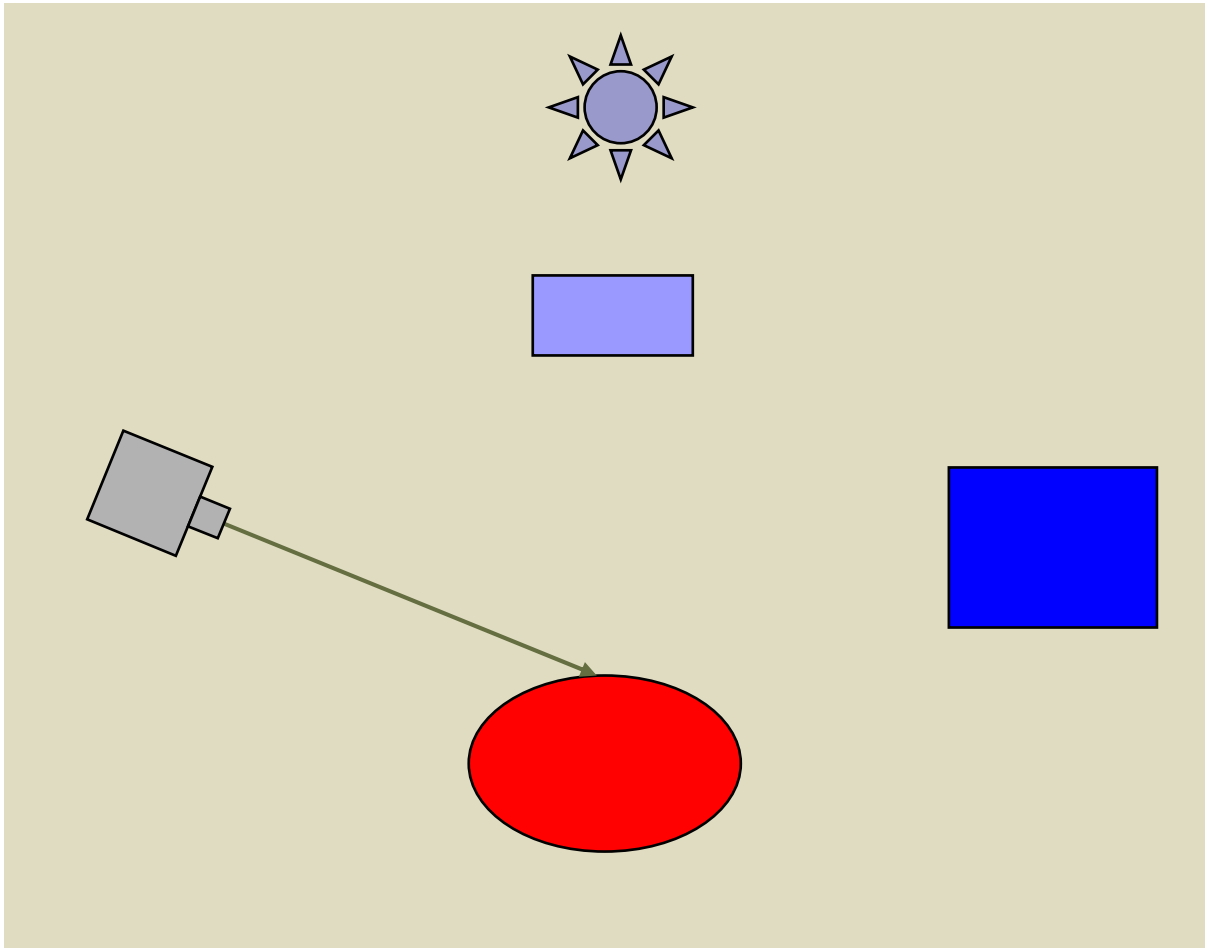


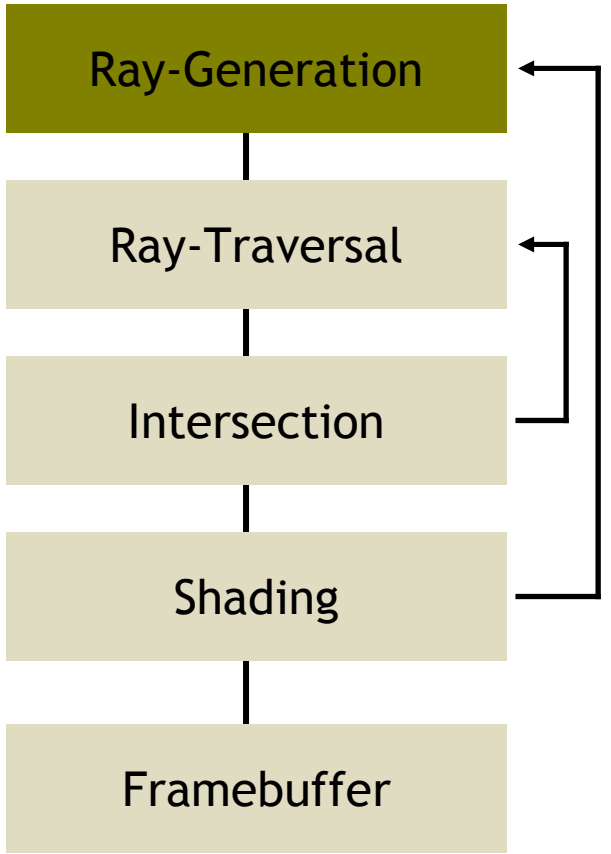
I passi dell'algoritmo [Introduction to Real-Time Ray Tracing http://www.openrt.de/Siggraph05/UpdatedCourseNotes/course.php]



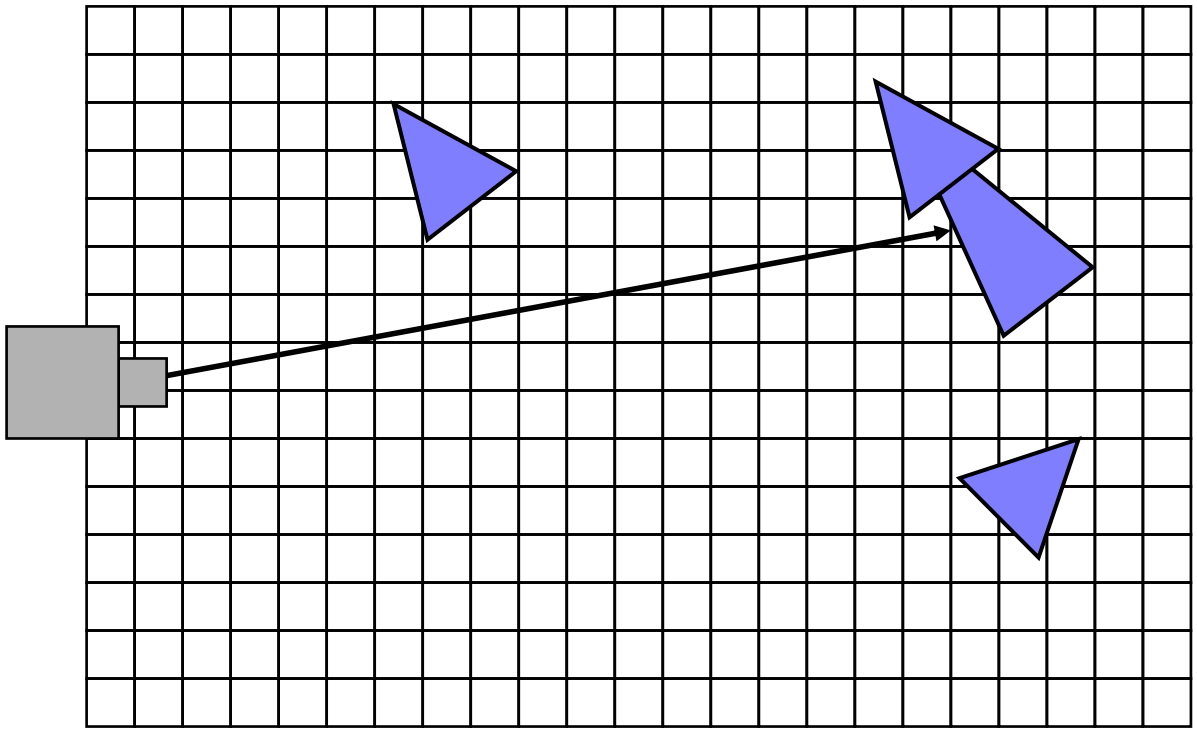


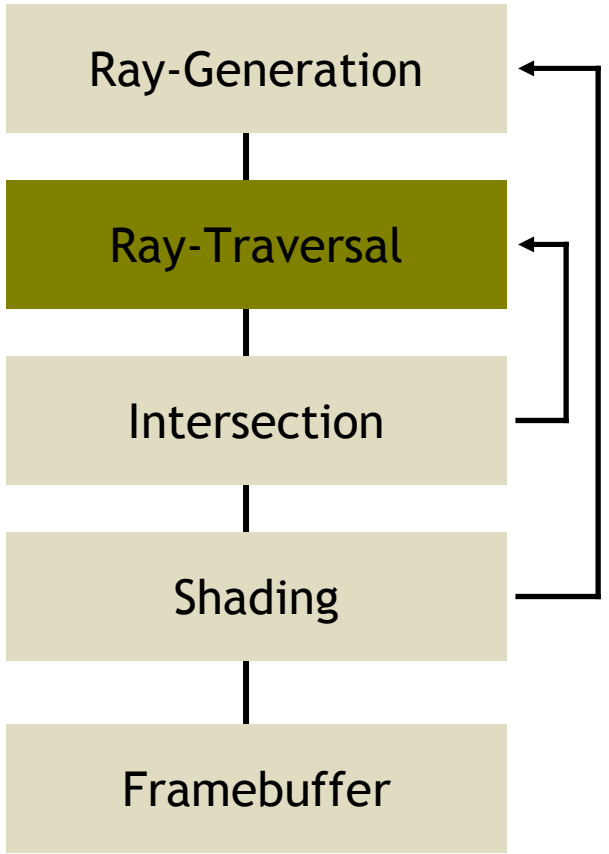
I passi dell’algoritmo



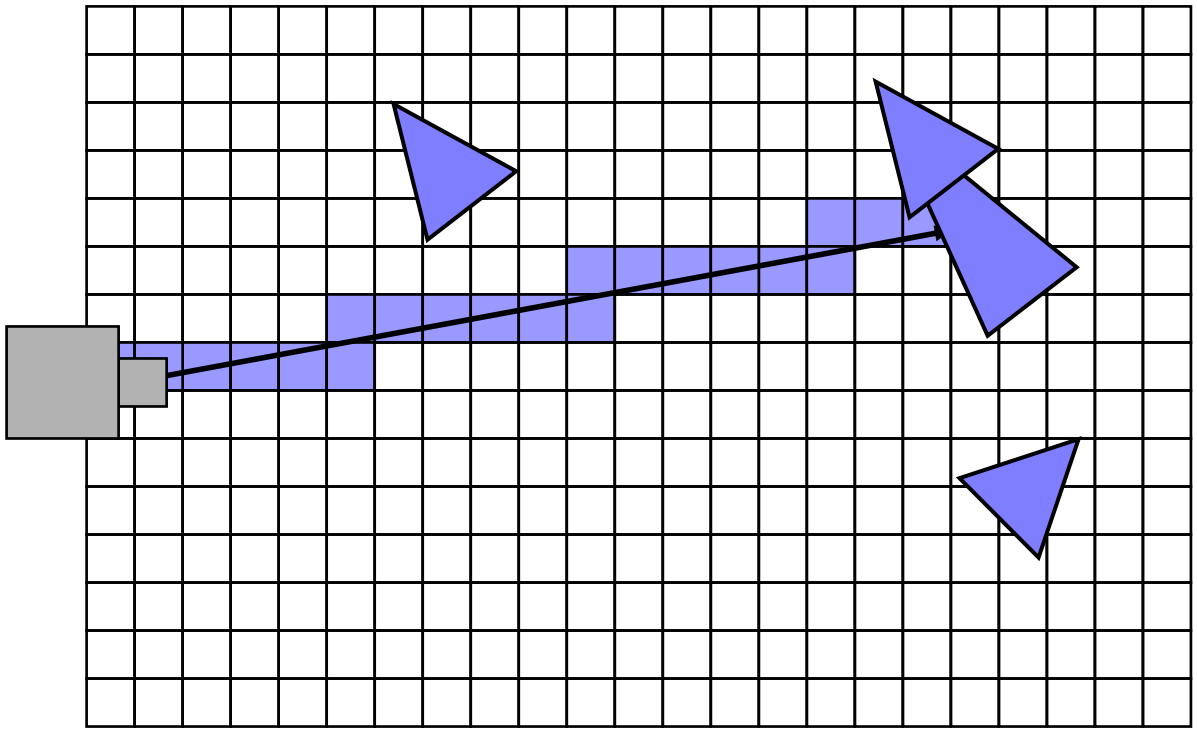


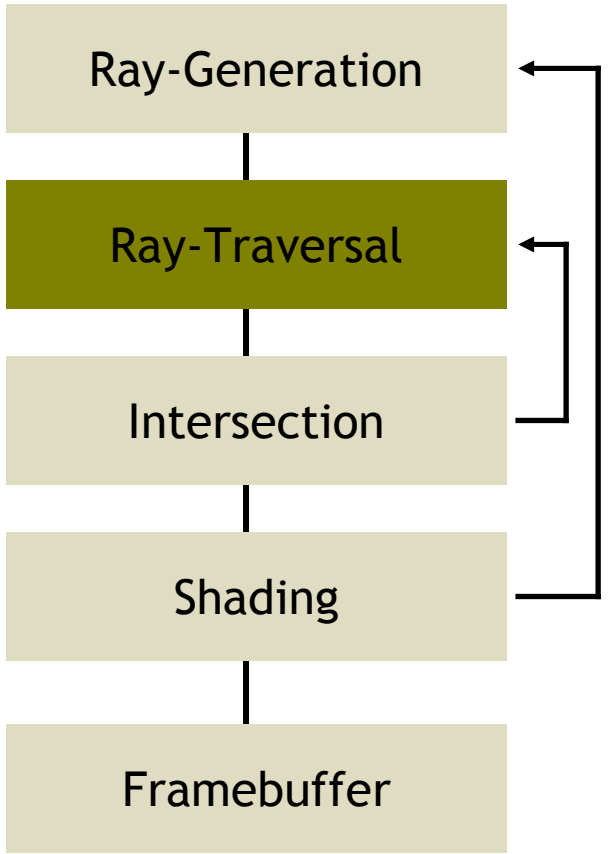
I passi dell'algoritmo: Traversal



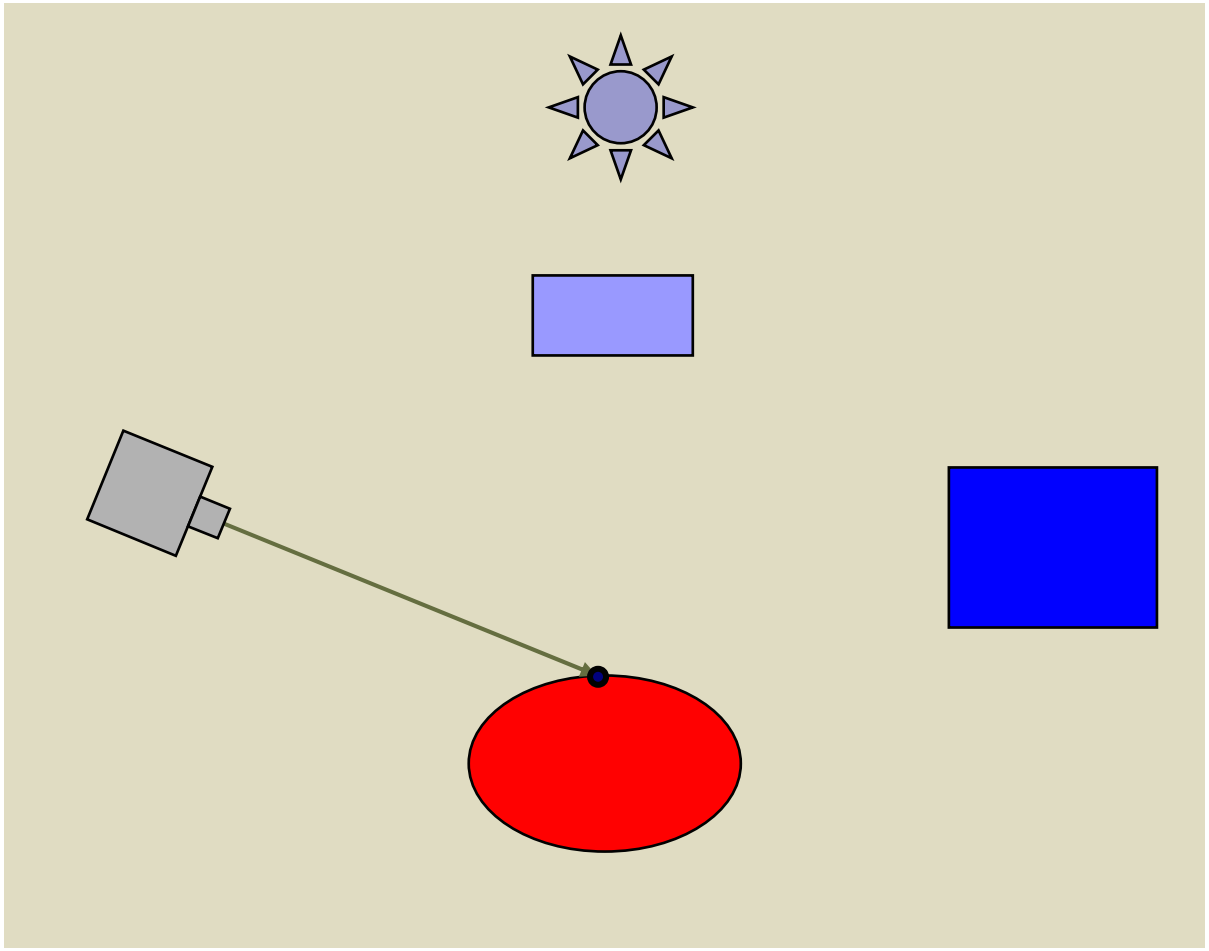


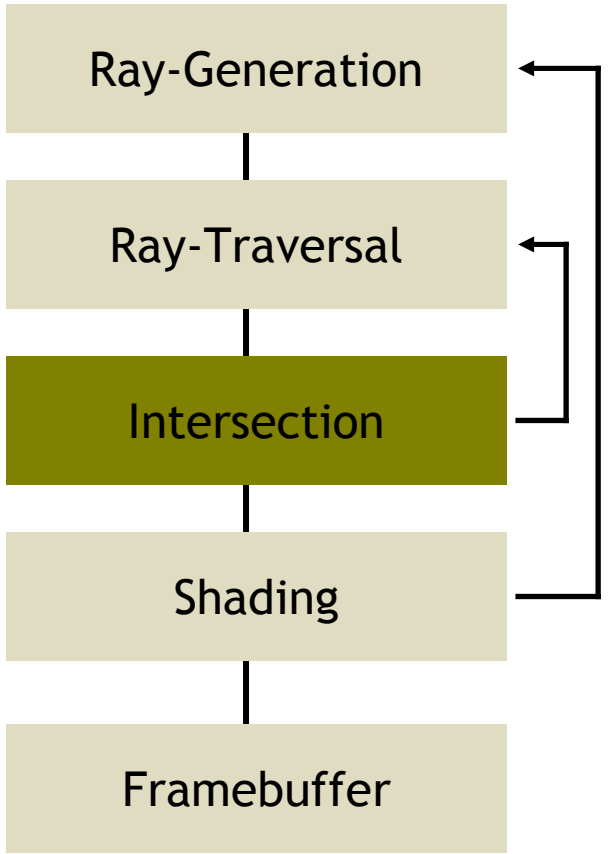
I passi dell'algoritmo: Traversal



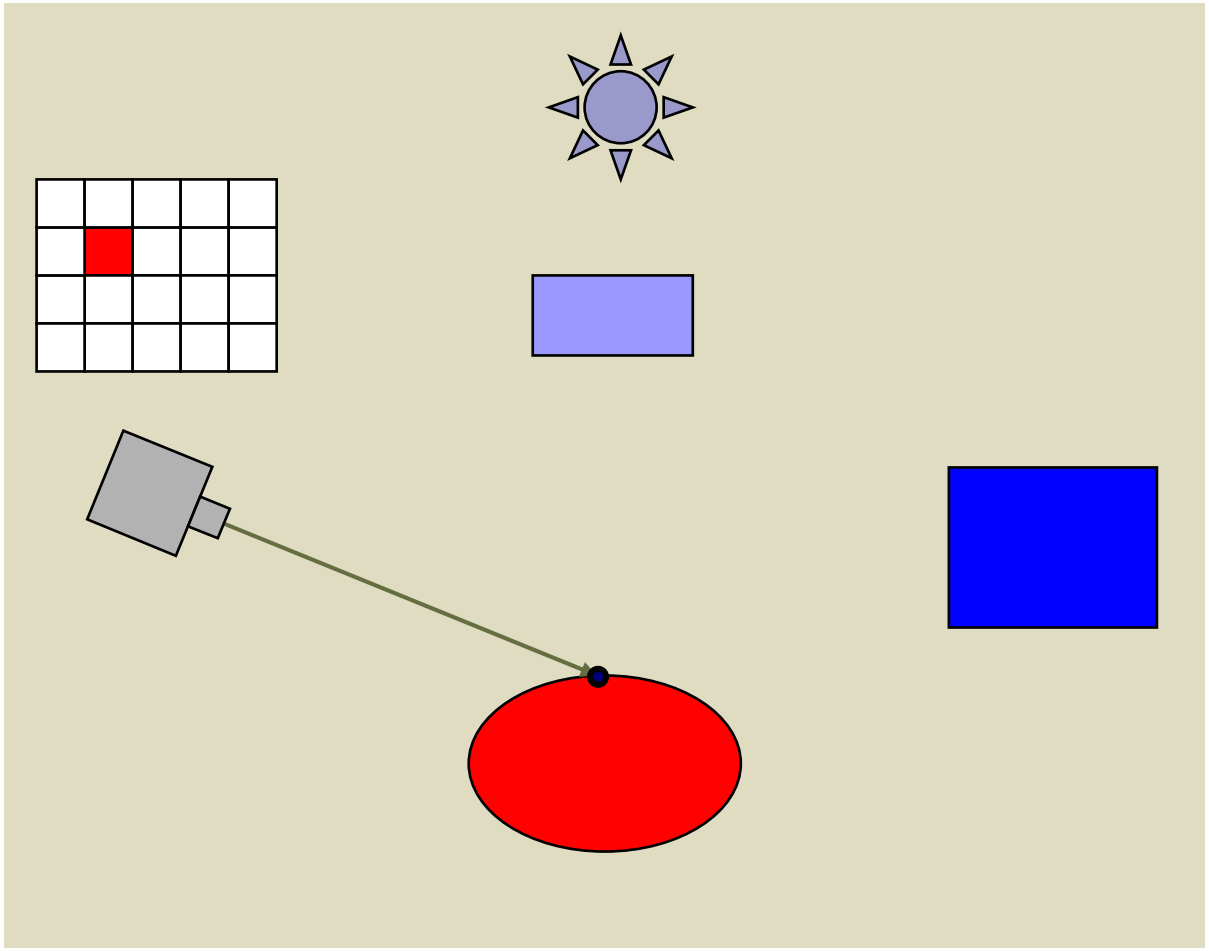


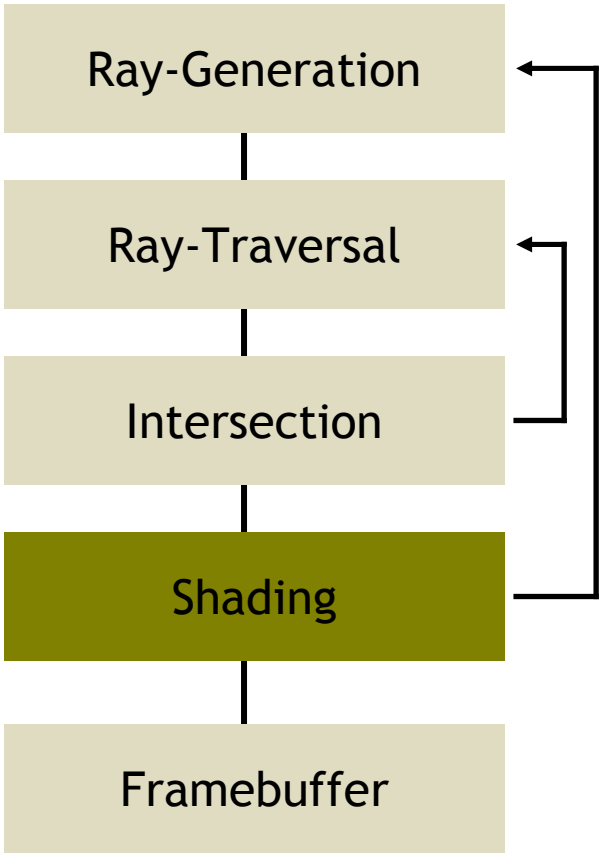
I passi dell’algoritmo



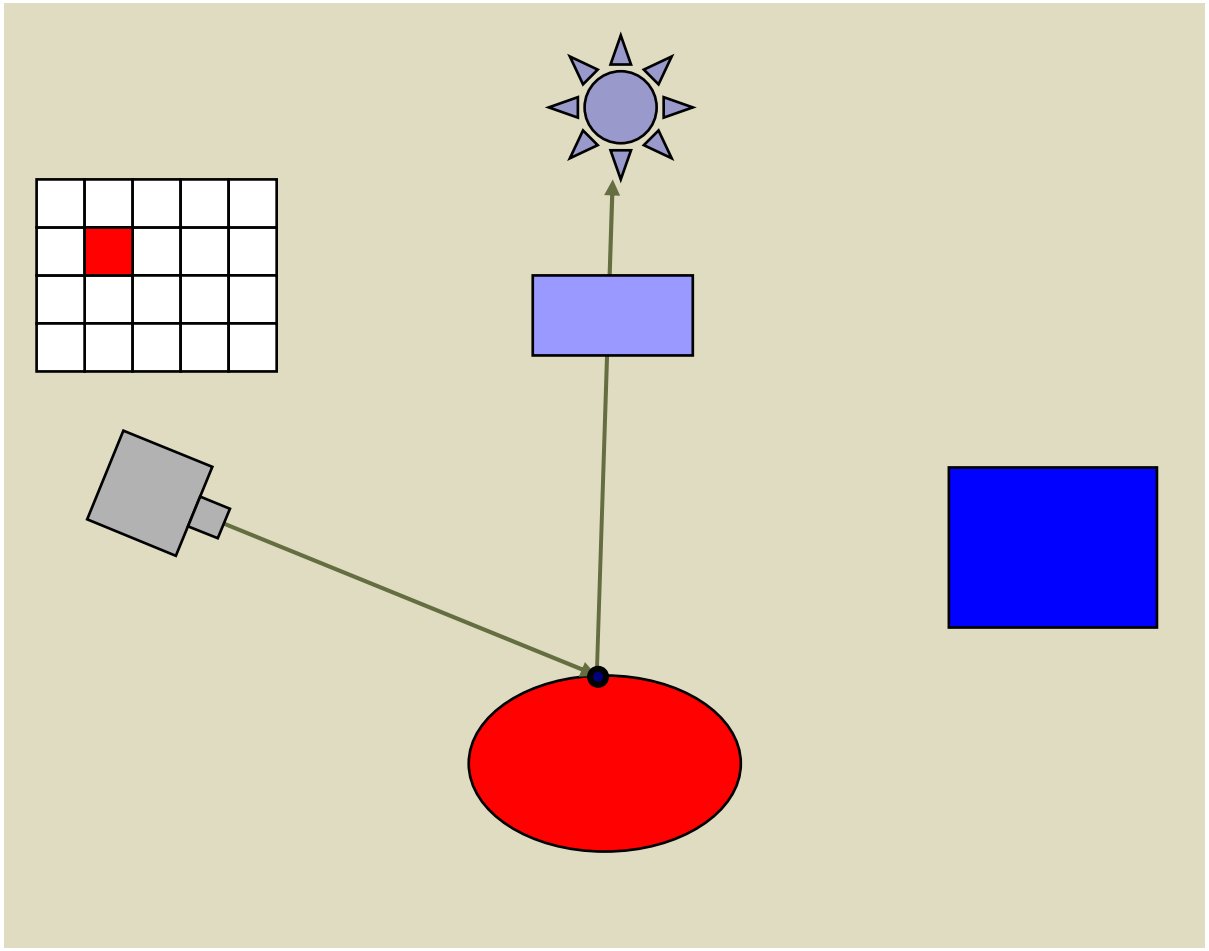


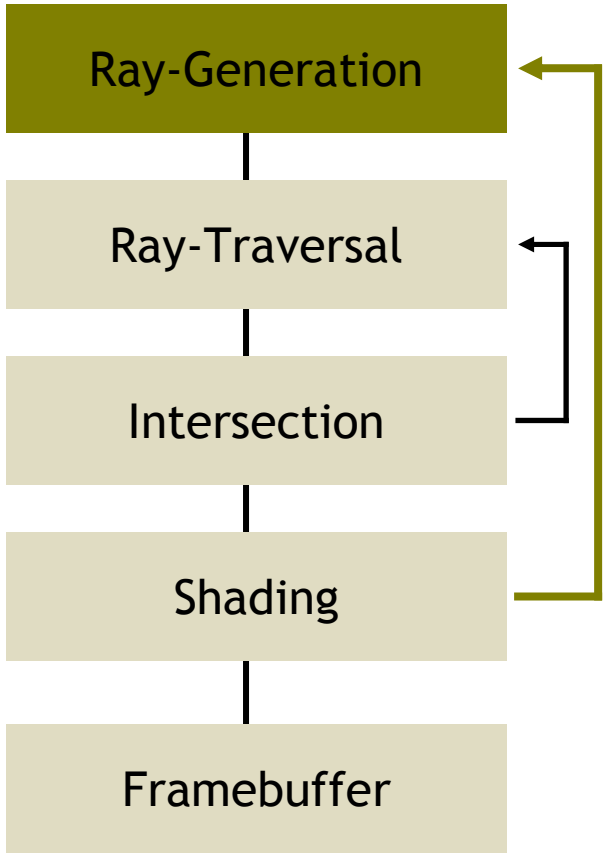
I passi dell’algoritmo:



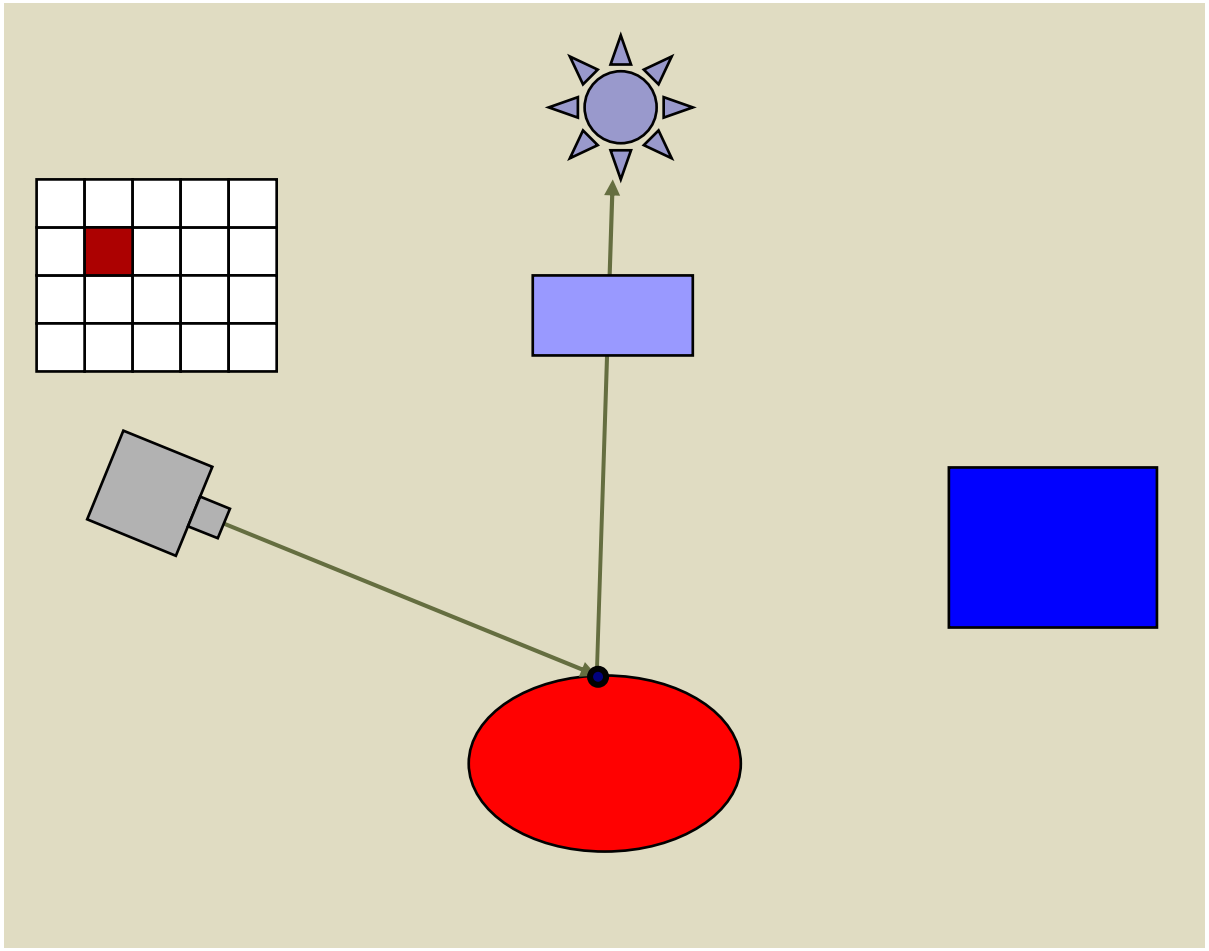


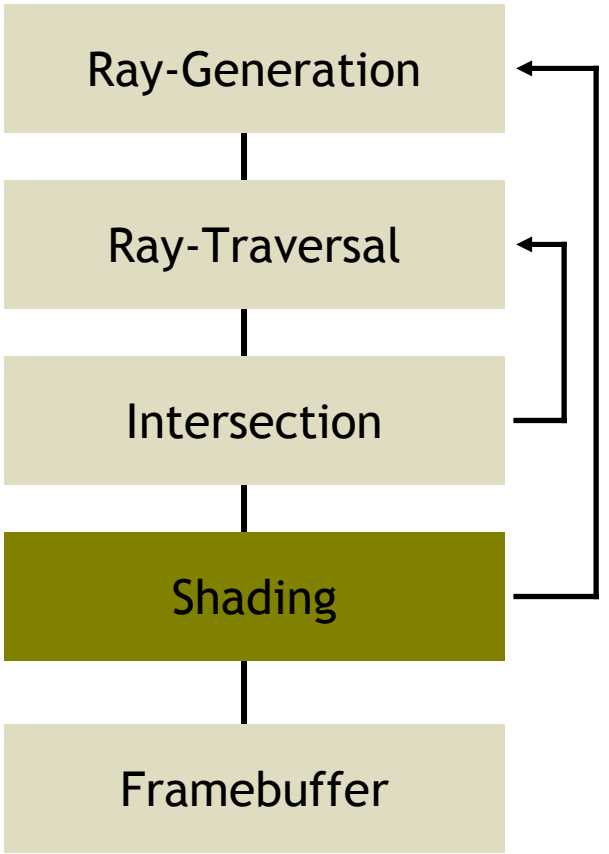
I passi dell’algoritmo:



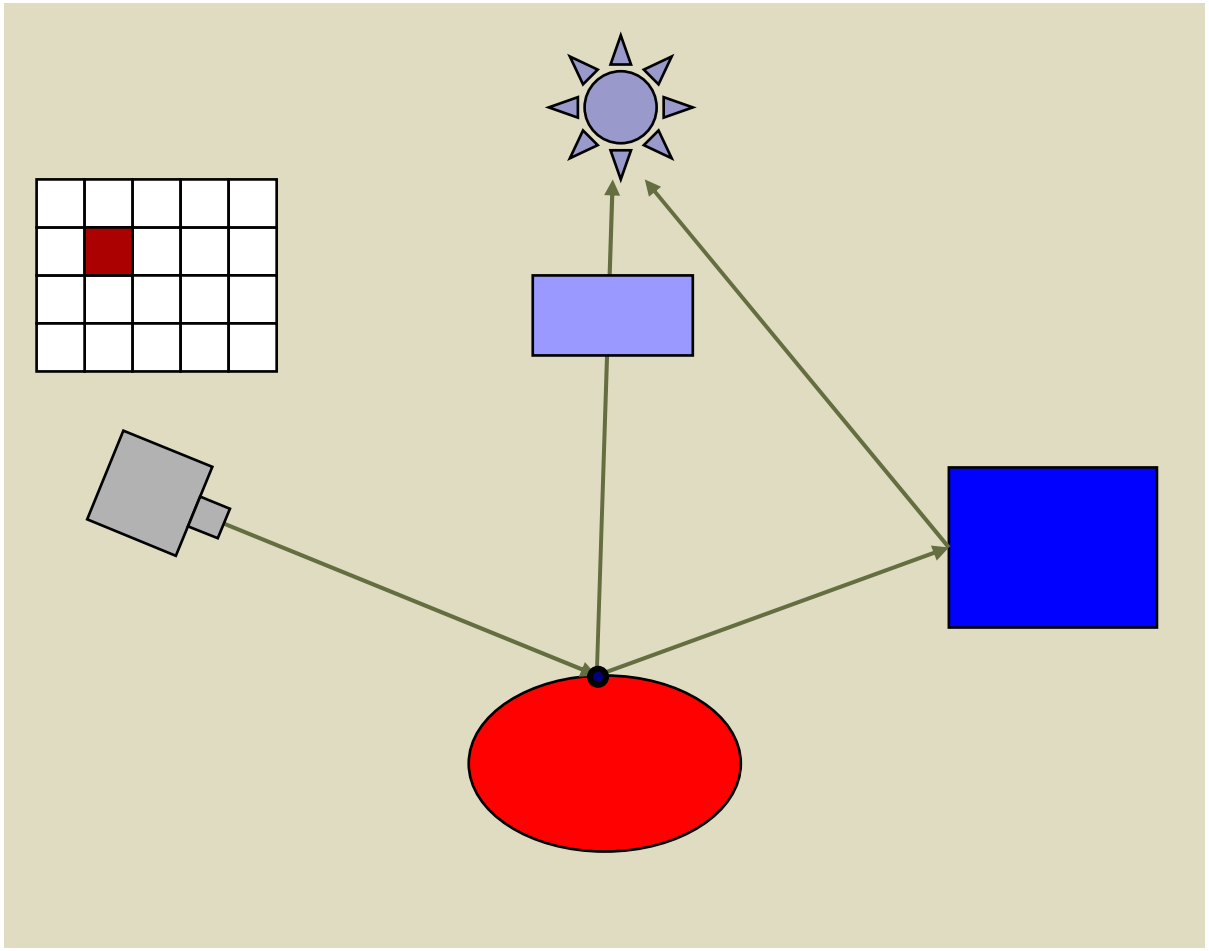


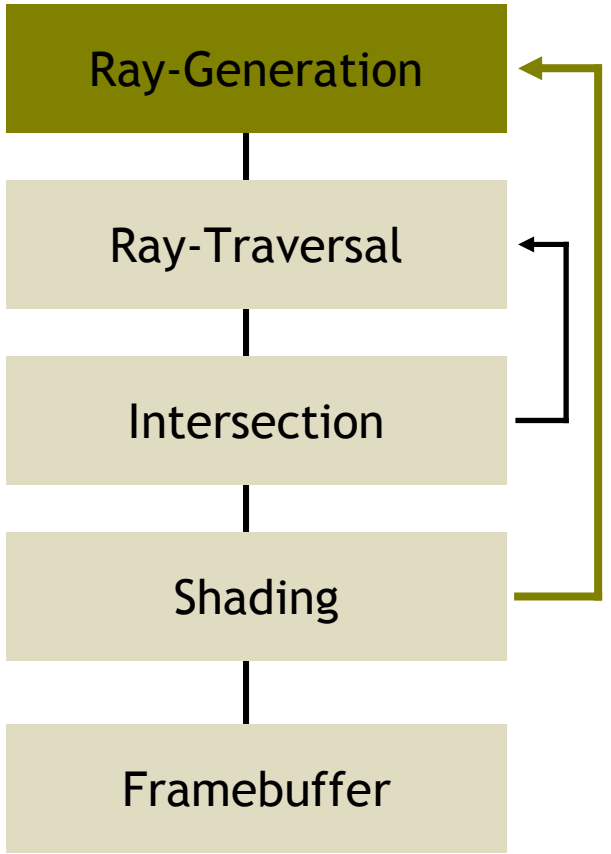
I passi dell’algoritmo:



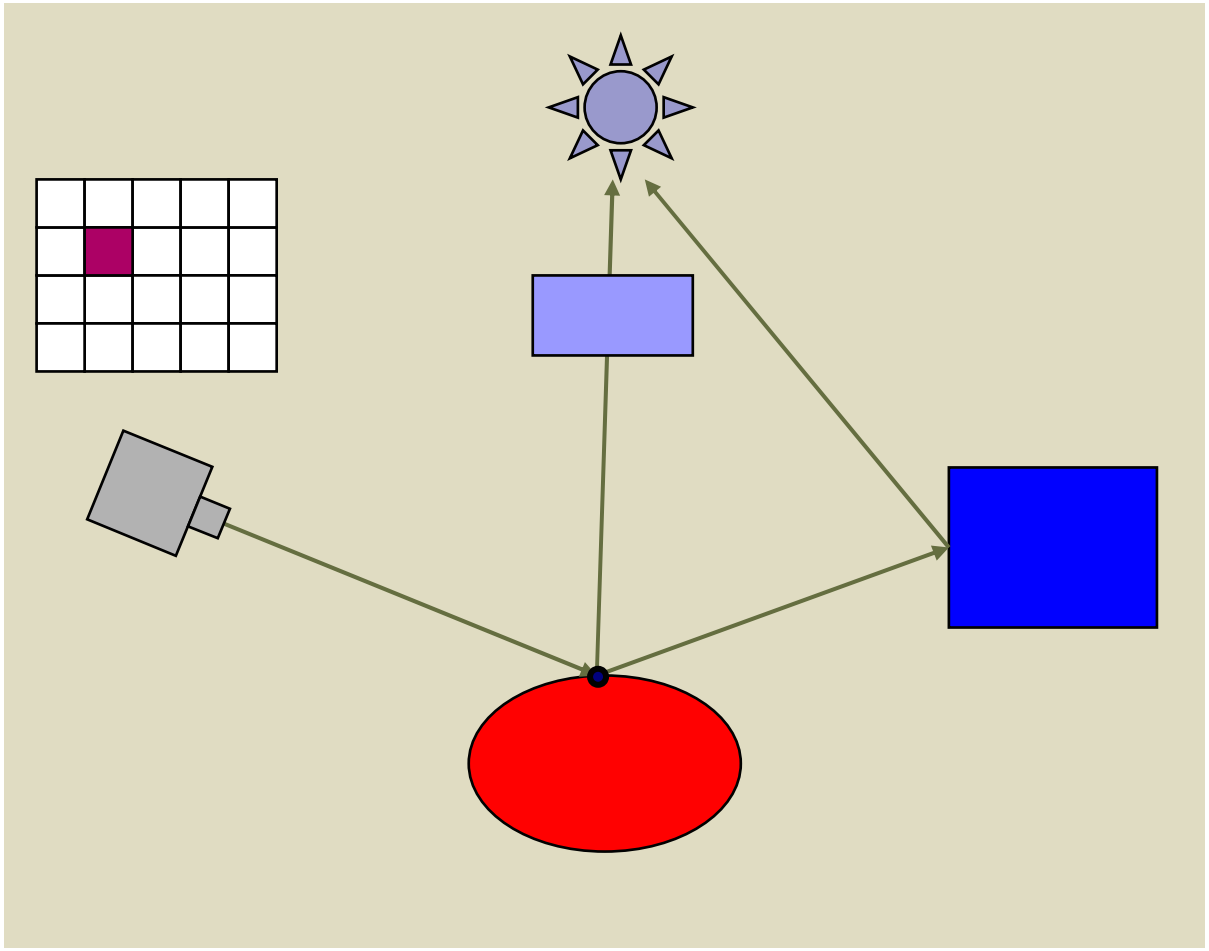


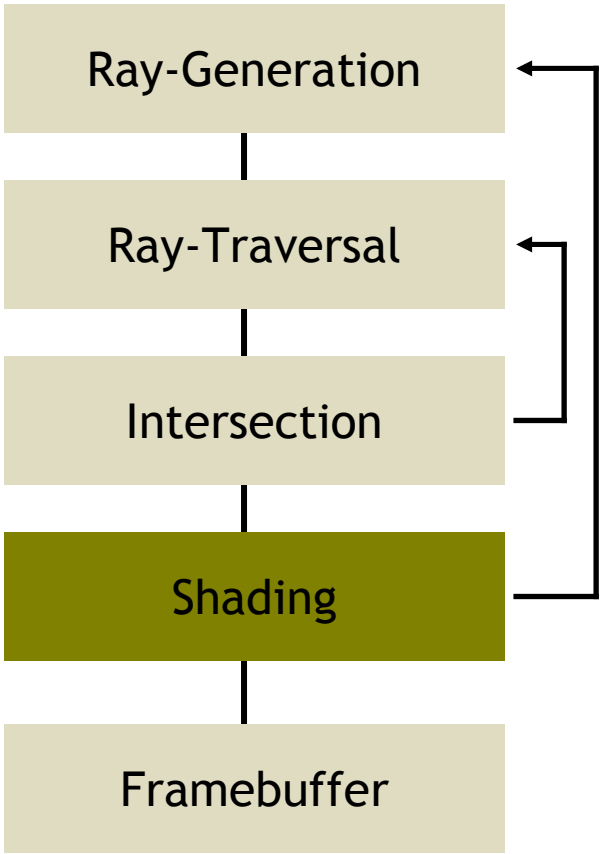
I passi dell'algoritmo:



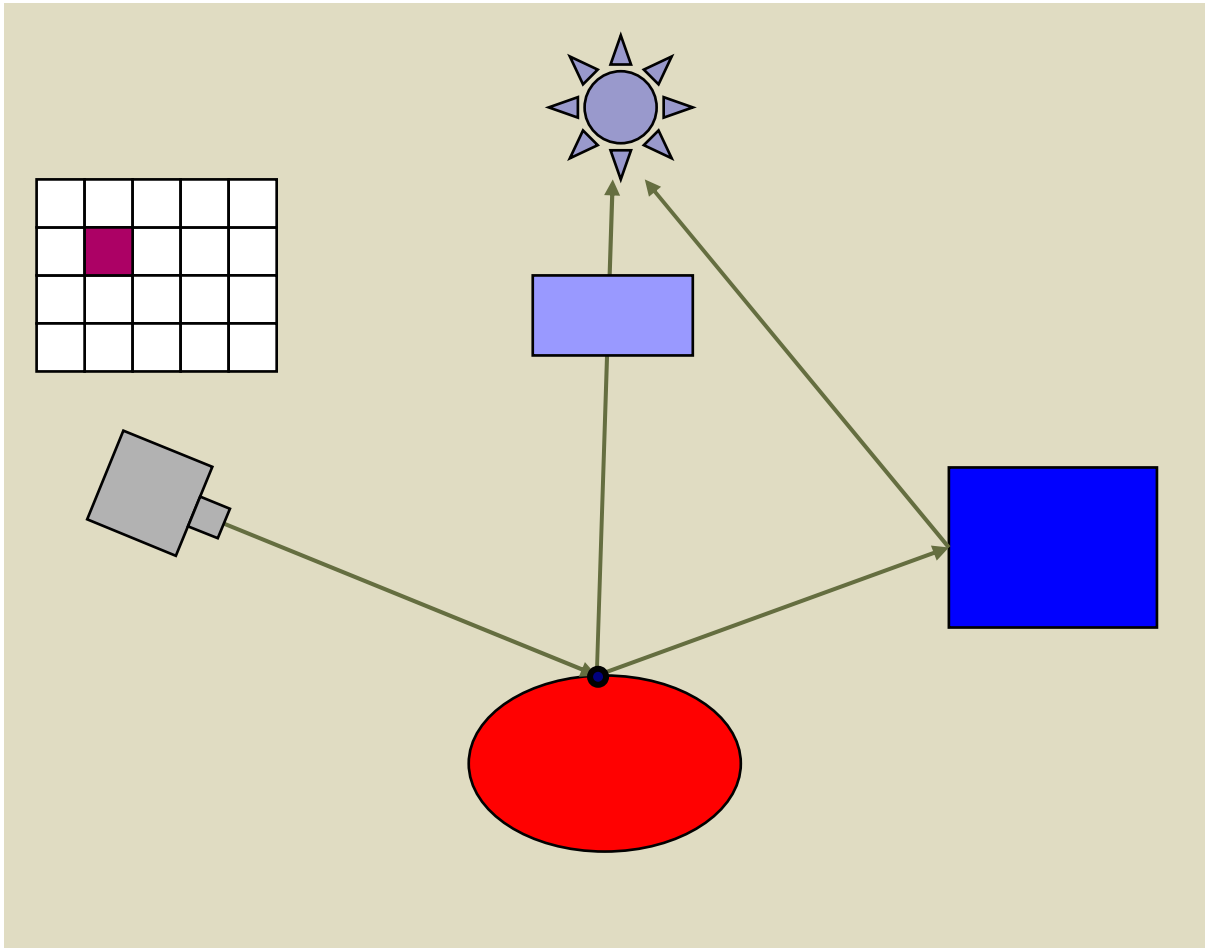


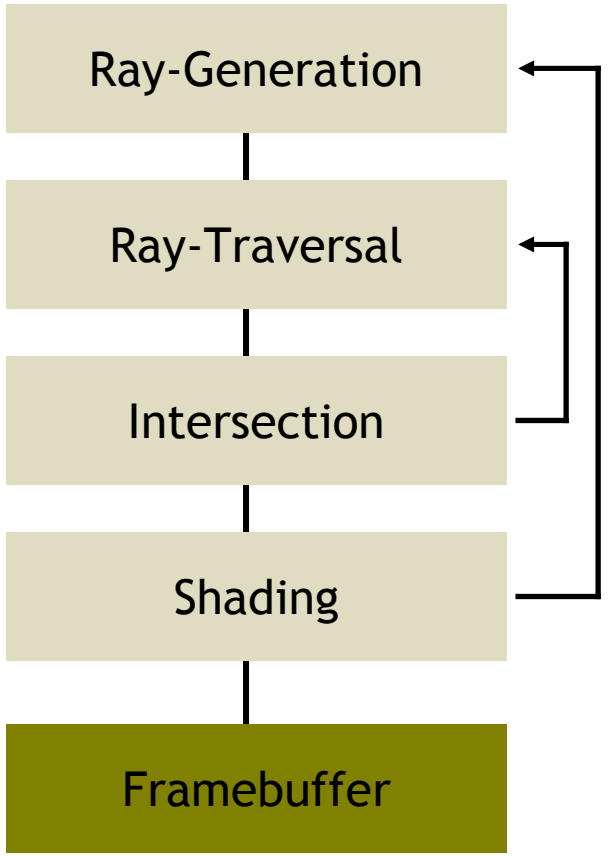
I passi dell’algoritmo:



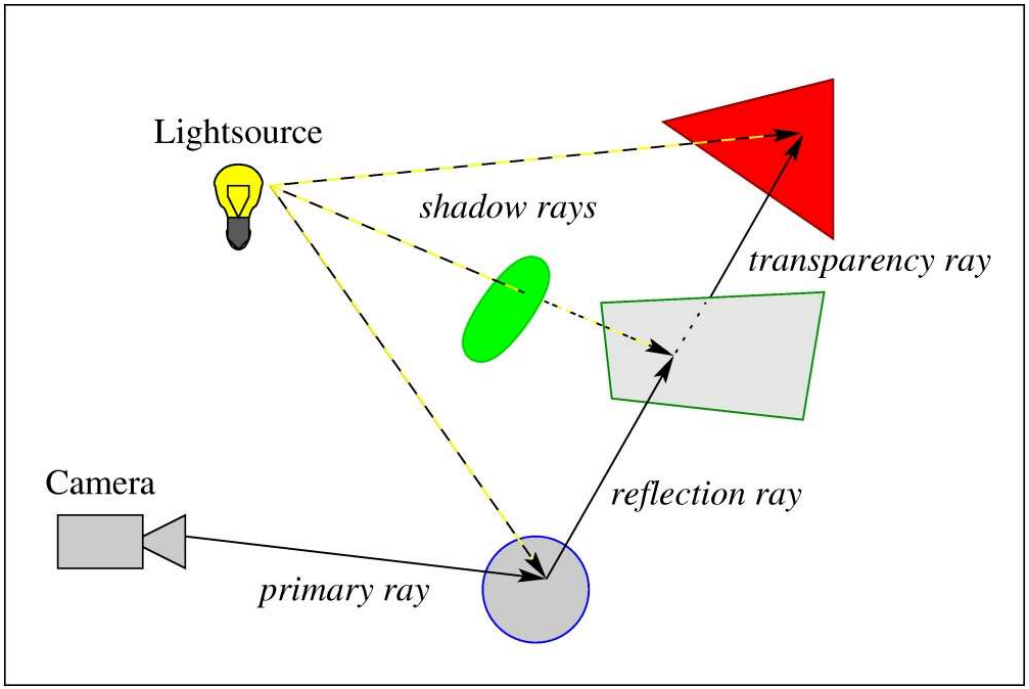


I passi dell'algoritmo:





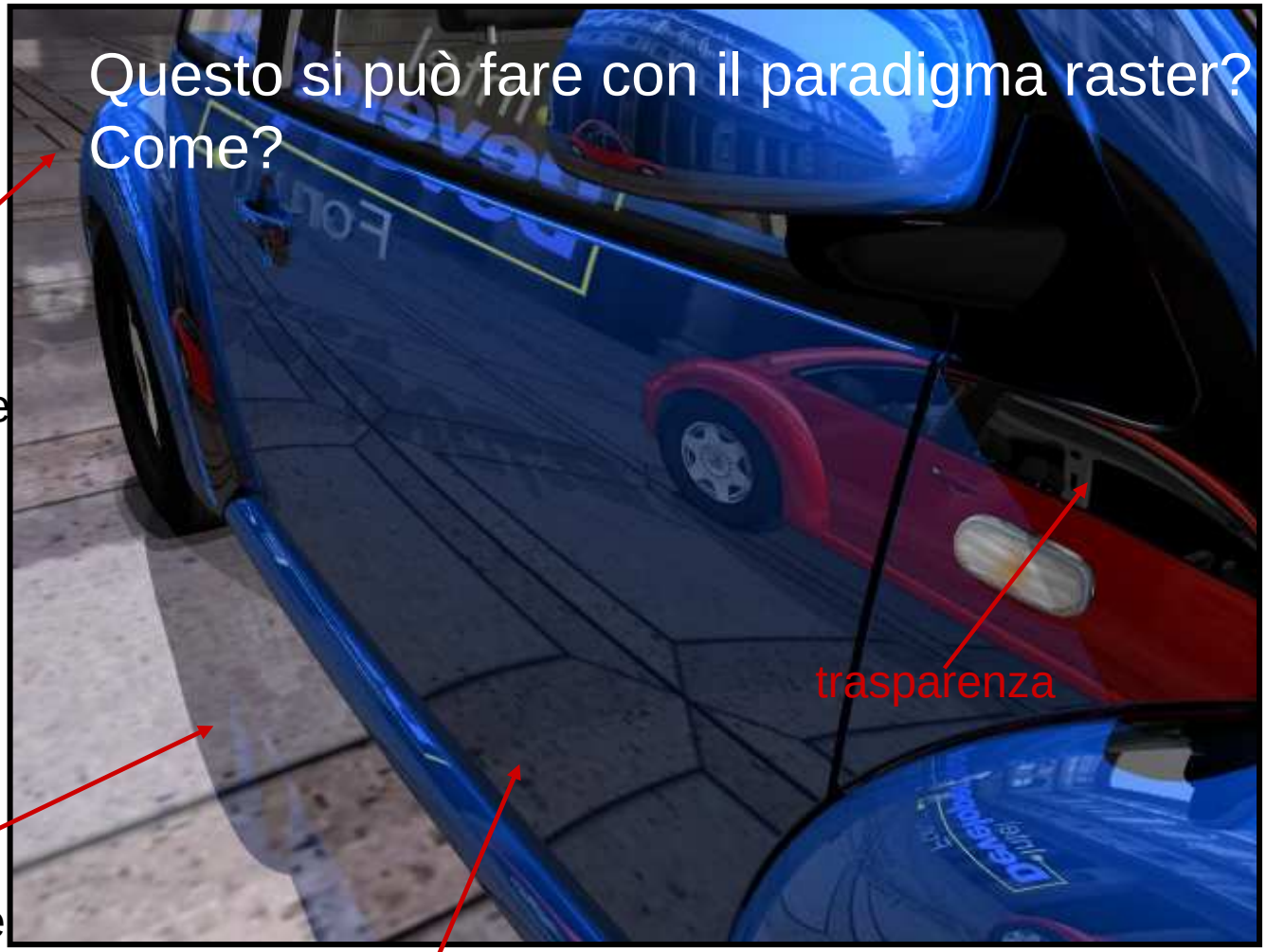
Le proprietà



- Effetti Globali
- Parallelizzabile
- Automatico (cioè?)
- Costo per pixel
- Efficiente (dipende)

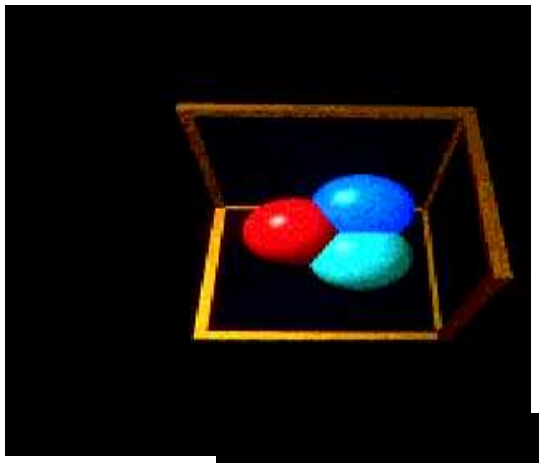
! Per ora le schede grafiche sono pensate per rasterization !

Benefici del Ray-Tracing

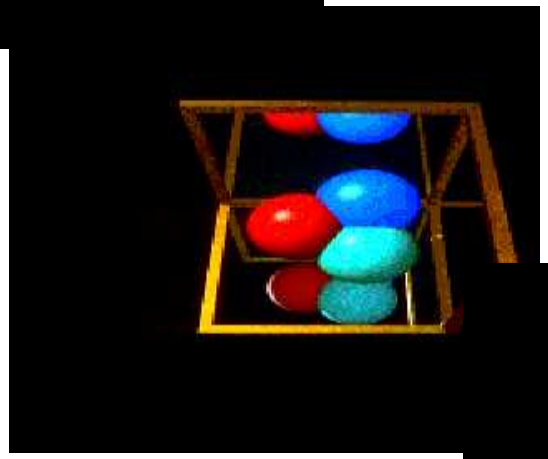


Benefici del Ray-Tracing

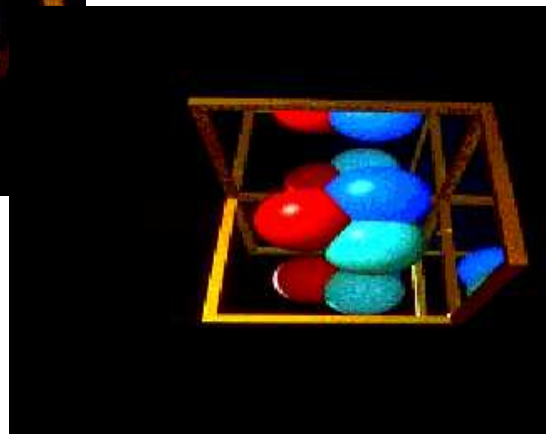
Quanti rimbalzi?



Solo primary ray



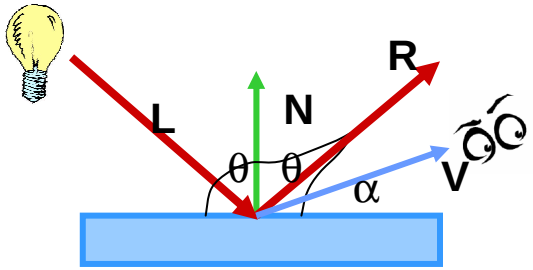
Un rimbalzo

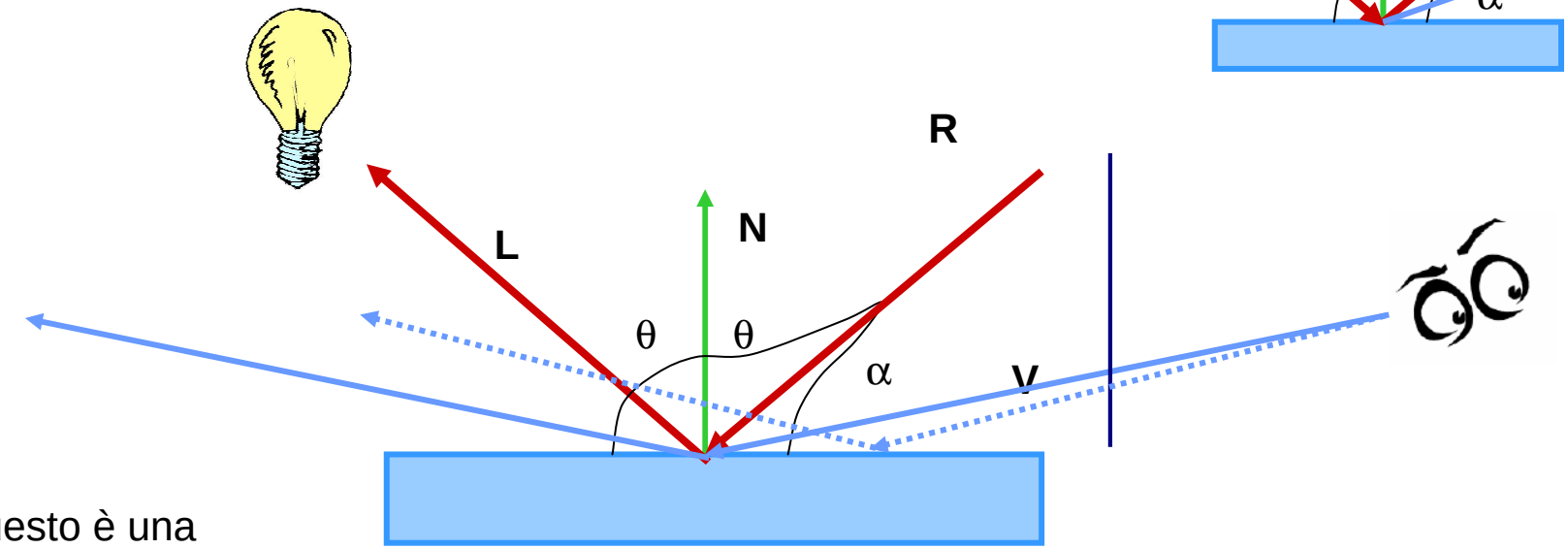


Due rimbalzi

Shading

- Memento: componente speculare





Questo è una caratteristica della luce

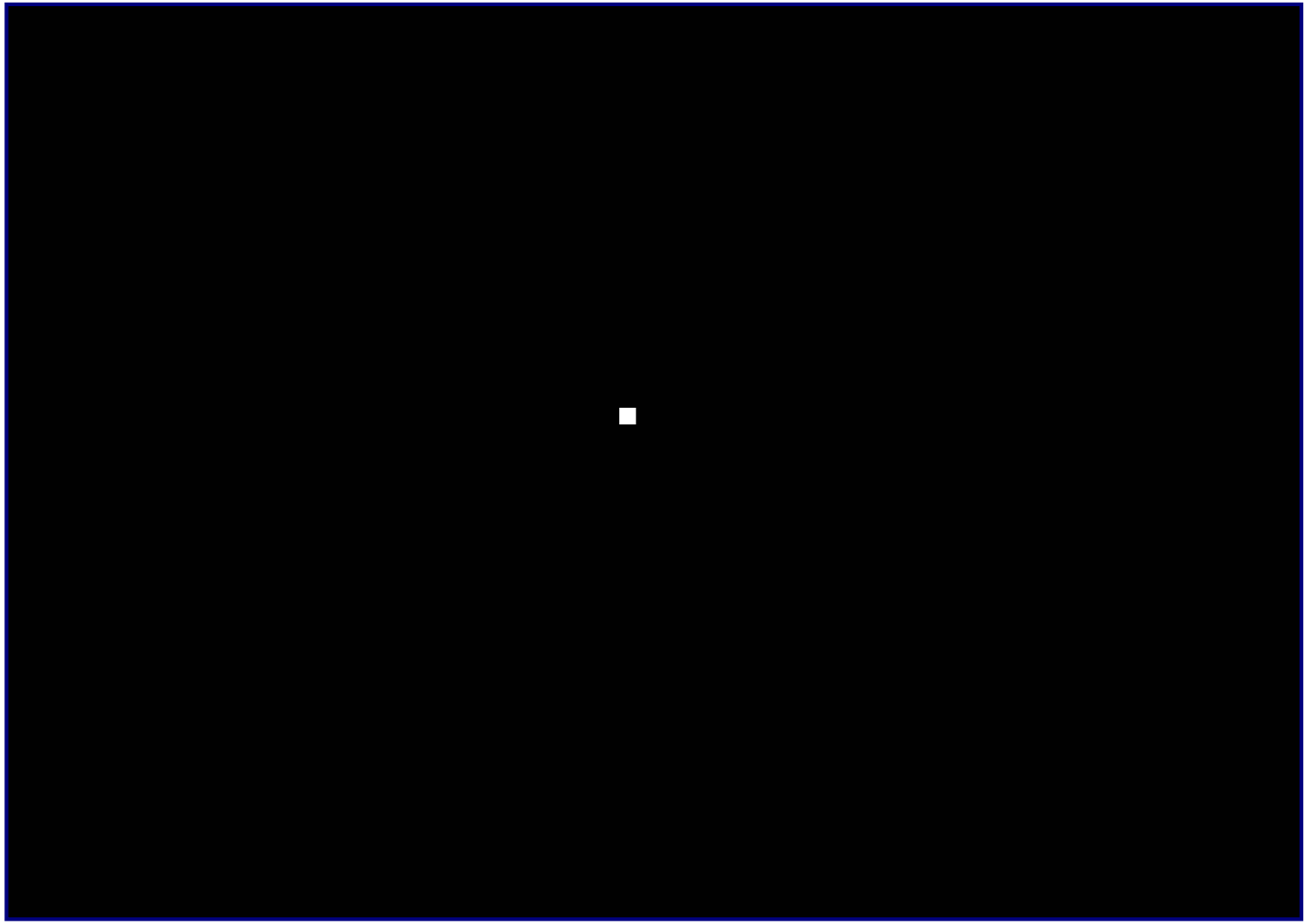
$$I_{spec} = I_{luce\ spec} \cdot k_{materiale\ spec} \cdot \cos^{\infty ?} \alpha$$

Questo è una caratteristica del materiale

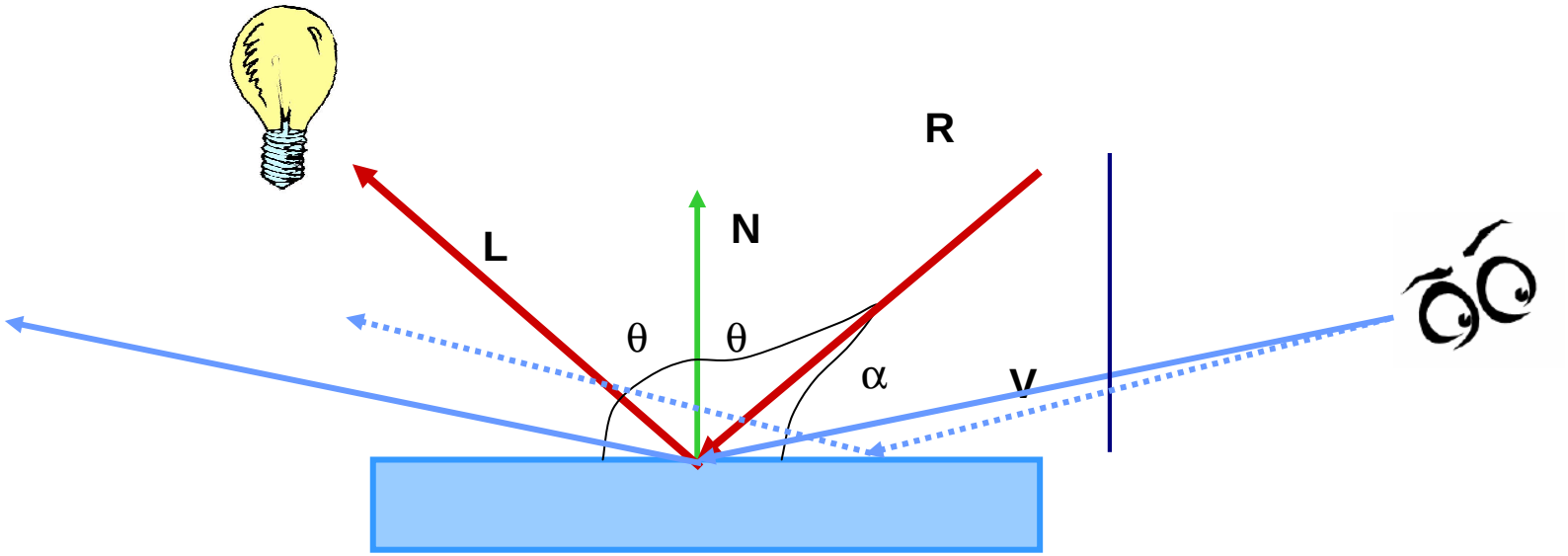
Come viene il rendering?

Così:

Ma potremmo avere molta fortuna:

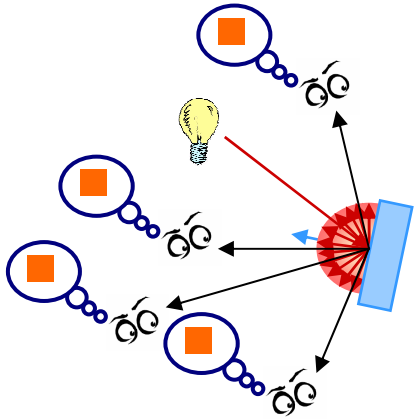


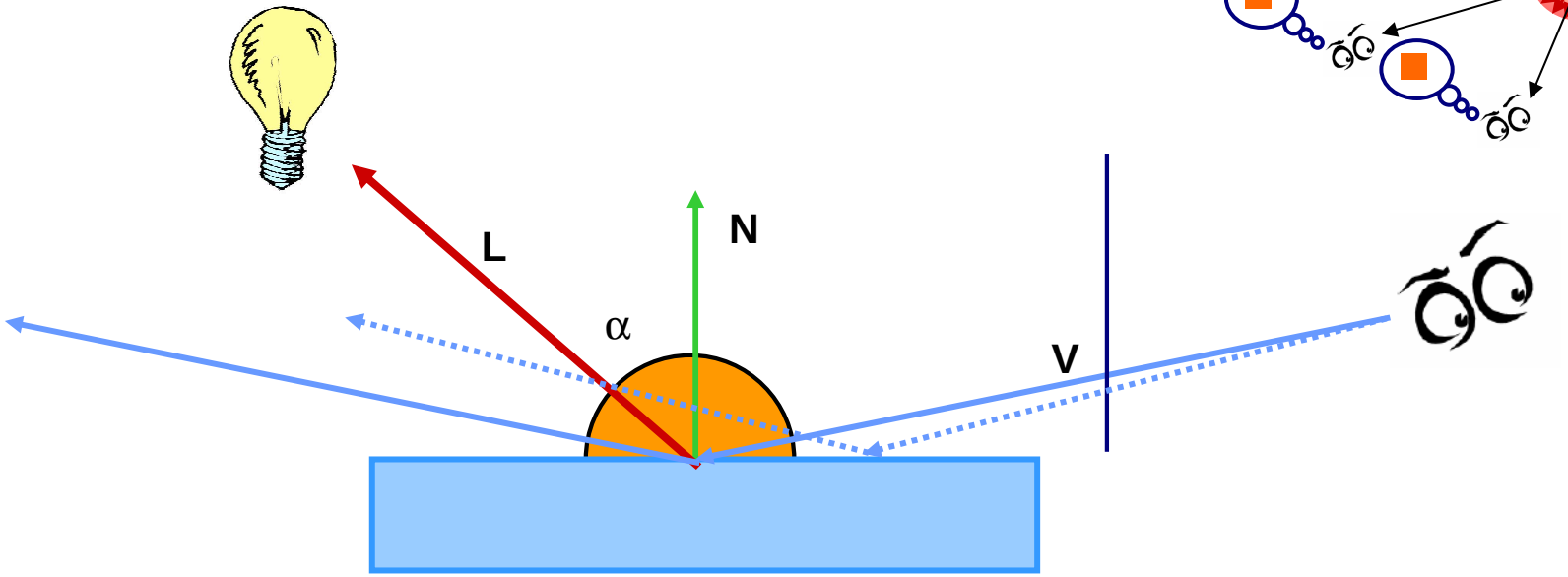
Shading: componente speculare



- All’ultimo passo serve un **modello locale** (ad esempio il modello di Phong)

Shading: componente diffusa

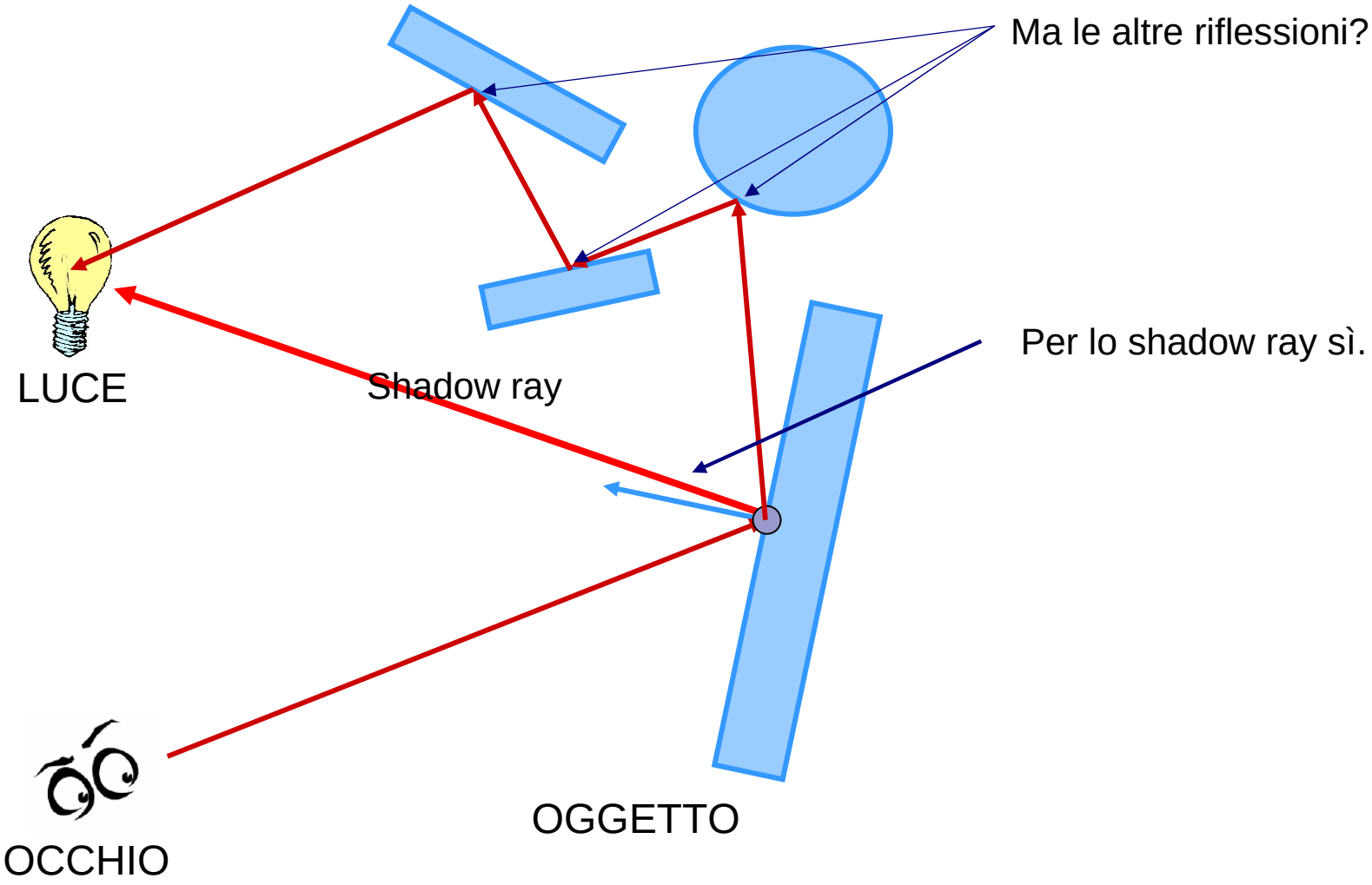




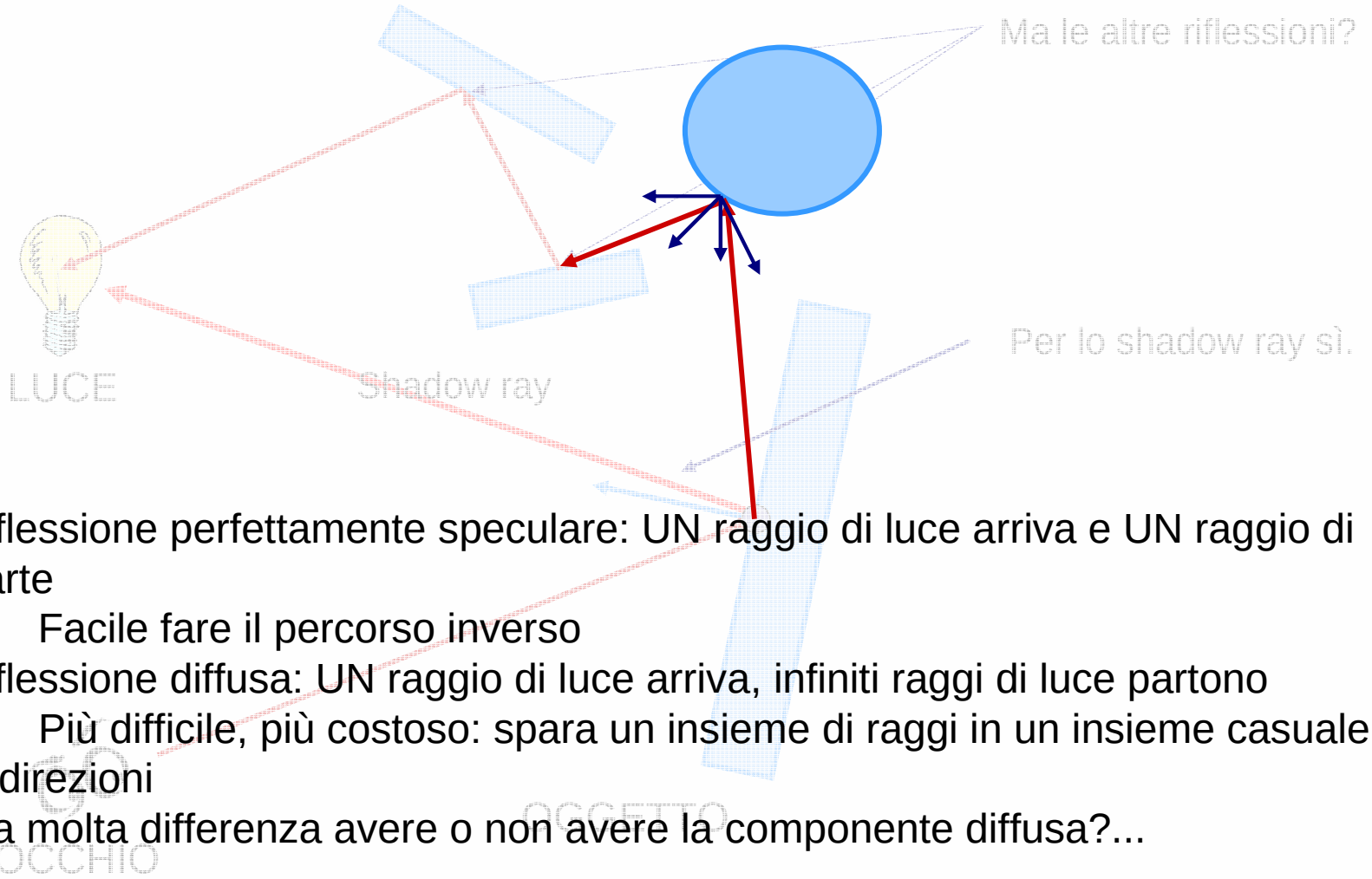
$$I_{diff} = I_{luce\ diff} \cdot k_{materiale\ diff} \cdot \cos\alpha$$

- Funziona allo stesso modo?

Shading: componente diffusa

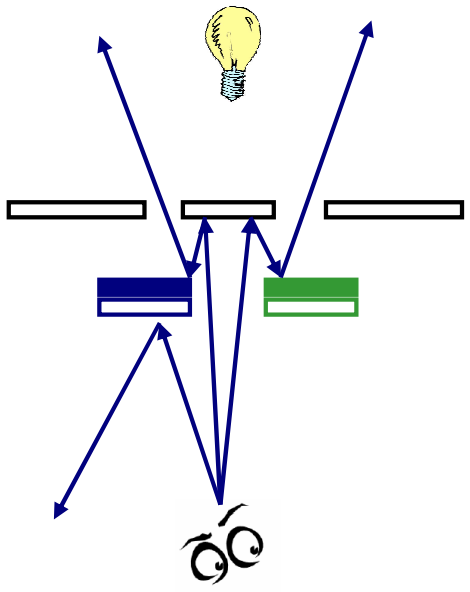


Shading: componente diffusa



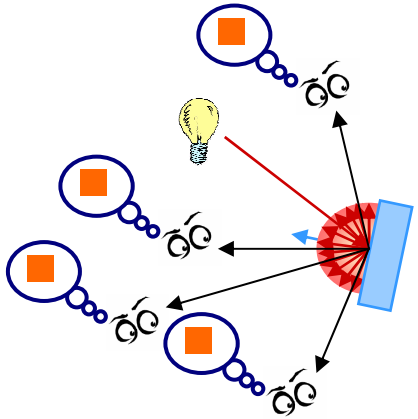
- Riflessione perfettamente speculare: UN raggio di luce arriva e UN raggio di luce parte
  - Facile fare il percorso inverso
- Riflessione diffusa: UN raggio di luce arriva, infiniti raggi di luce partono
  - Più difficile, più costoso: spara un insieme di raggi in un insieme casuale di direzioni
- Fa molta differenza avere o non avere la componente diffusa?...

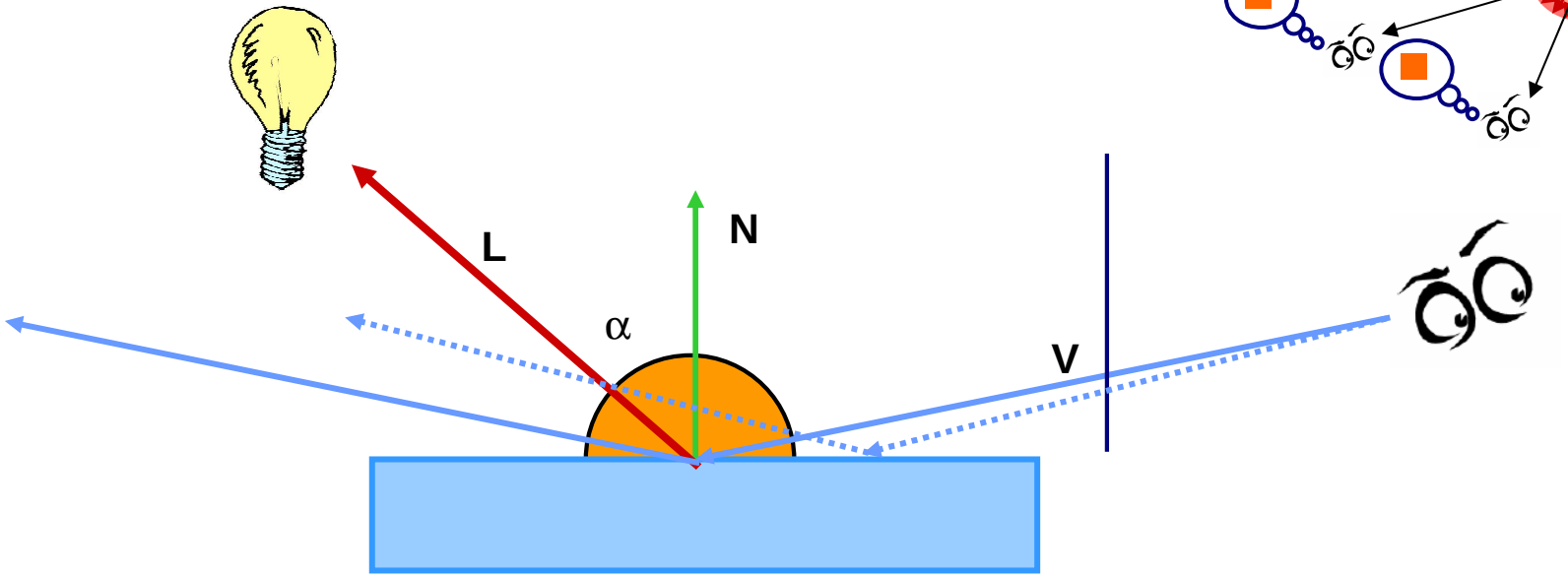
Color bleeding



È un caso pessimo, ma è la ragione per la quale le immagini migliori composte con il ray tracing comprendono sempre superfici liscie, materiali semitrasparenti etc..

Shading: componente diffusa





$$I_{diff} = I_{luce\ diff} \cdot k_{materiale\ diff} \cdot \cos\alpha$$

- All'ultimo passo serve un **modello locale** (ad esempio il modello di Phong)

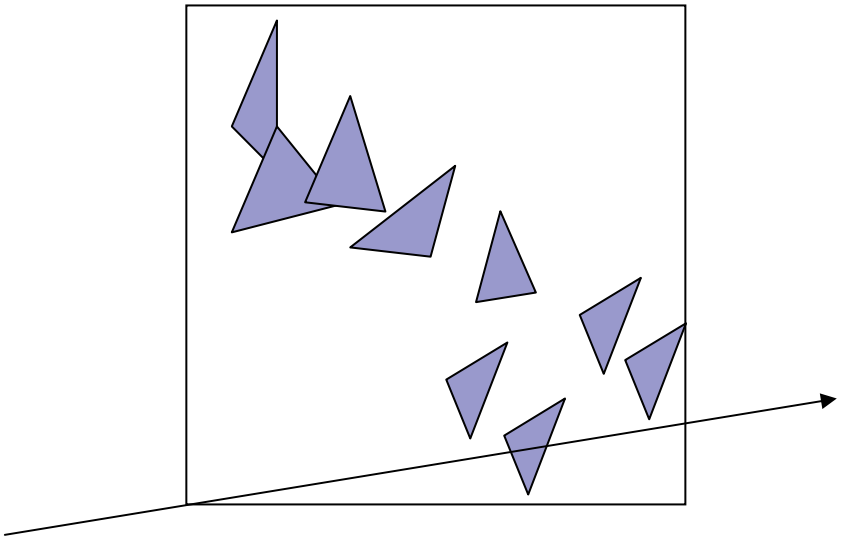
Gli ingredienti

- Uno fondamentale: “Data la descizione della scena, determinare in modo EFFICIENTE l’intersezione di un raggio con la scena”
  - Primary ray
  - Secondary ray
  - Shadow ray
  - Trasparency ray
- Nota:
  - Su una finestra 800x600
  - Con un solo raggio per pixel (caso base)
  - = 480K raggi per frame
  - …solo per i primary rays

Intersezione raggio - scena

Come trovo l'intersezione di un raggio con la scena (che supponiamo composta di *n* triangoli) ?

Per ogni triangolo, calcolo l'intersezione (eventuale) con il raggio. Il risultato è l'intersezione più vicina al punto di partenza del raggio.



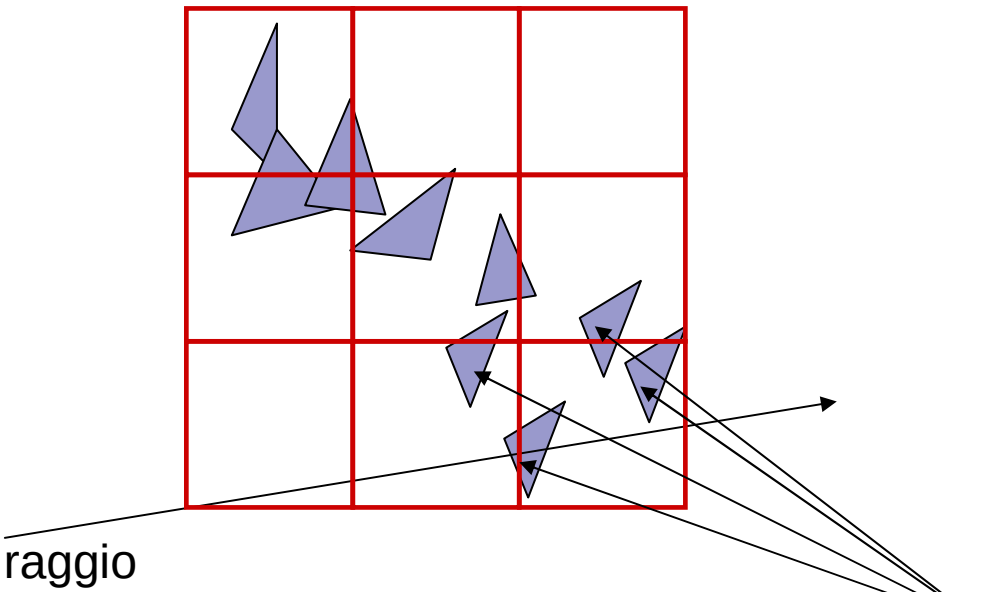

```
RaySceneIntersection(r){
  closest = infinite;
  closest_p = NULL;
  for p in triangles
  if( r intersects p){
    t = intersection(r,p);
    if(t < closest){
      closest = t;
      closest_p = p;
    }
   }
}
```

Peggio di così non si può fare!

Strutture di indicizzazione spaziale

Idea: cerchiamo di ridurre il numero di test di intersezione raggio-triangolo

1. Immagino una griglia uniforme nello spazio
2. Ad ogni cella associo l'insieme di primitive che la intersecano
3. **Rasterizzo** il raggio sulla griglia: per ogni cella rasterizzata, eseguo **l'intersezione** tra il raggio e le primitive associate



Già meglio…

```
RaySceneIntersection(r){
  cell = NextCell();
  while(cell != NULL){
  for p in cell
  if( r intersects p){
    t = intersection(r,p);
    break;
    }
    cell = NextCell();
  }
}
```

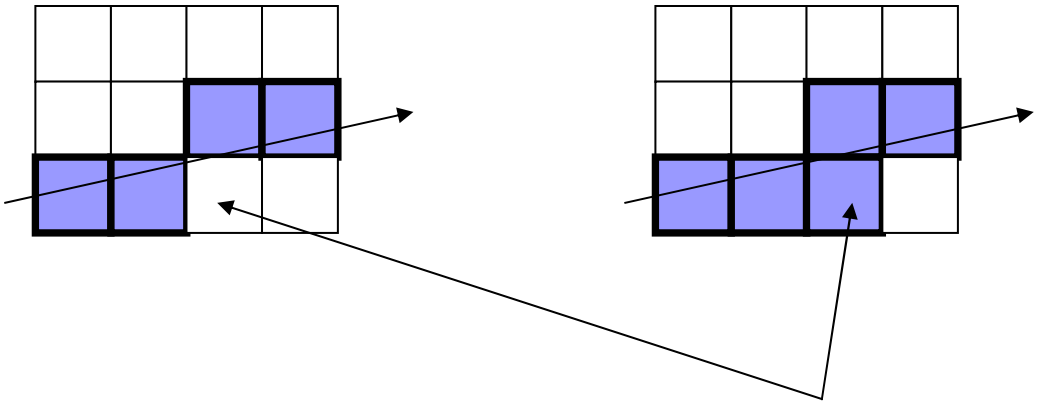
primitive effettivamente testate

DDA-3d: rasterizzazione in 3D

- È molto simile alla rasterizzazione dei segmenti in 2d
  - Simile, non uguale
  - Anche il problema non è uguale

Per disegnare la linea

Per il ray tracing



Per il ray tracing mi interessano **tutte**
le celle intersecate dal raggio

DDA-3d: rasterizzazione in 3D [Amanatides&Woo 87]

- Niente direzioni preferenziali (m<1, m>1 etc..)
- L'algoritmo si muove da un'intersezione con un bordo di una cella all'altra
- L'intersezione può essere con ognuno dei piani principali (XY,XZ,YZ)
- Forma parametrica del raggio

$$r = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \cdot t + \begin{bmatrix} cx \\ cy \\ cz \end{bmatrix}$$

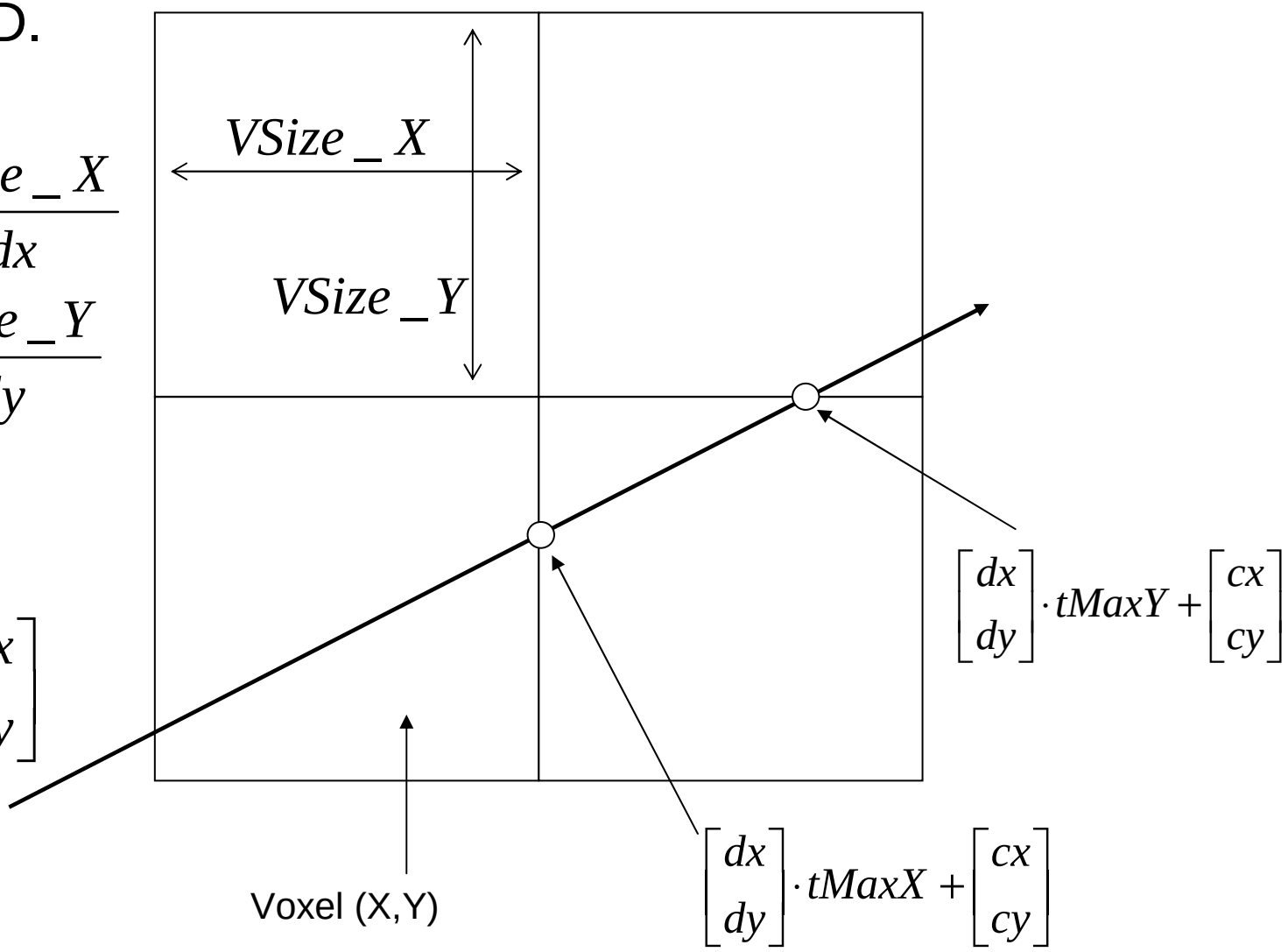
DDA-3d: rasterizzazione in 3D [Amanatides&Woo 87]

- Caso 2D.

$$tDeltaX = \frac{VSize\_X}{dx}$$

$$tDeltaY = \frac{VSize\_Y}{dy}$$

$$r = \begin{bmatrix} dx \\ dy \end{bmatrix} \cdot t + \begin{bmatrix} cx \\ cy \end{bmatrix}$$



DDA-3d: rasterizzazione in 3D [Amanatides&Woo 87]

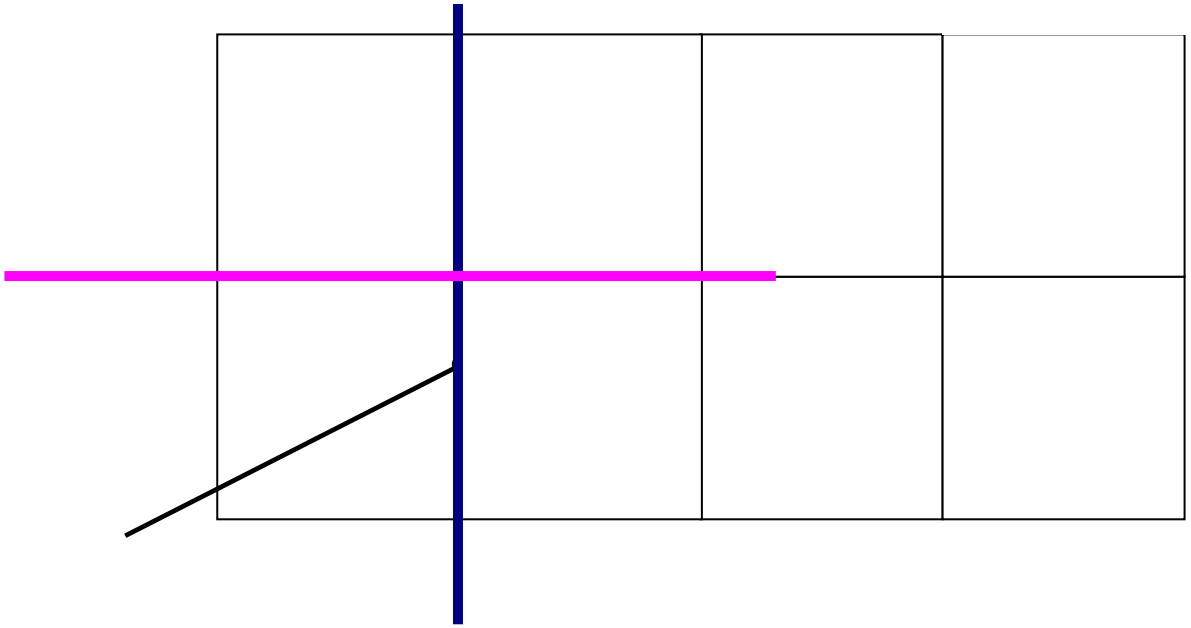
- Variabili:
  - X,Y: variabili intere che identificano il voxel corrente
  - tMaxX (tMaxY): valore massimo di t per il quale la coordinata X (Y) del voxel che contiene il punto rimane immutata
  - tDeltaX (tDeltaY): valore di cui incrementare t per spostare un punto lungo il raggio della dimensione del voxel in direzione X (Y)
- Inizializzazione di X,Y:
  - Voxel di partenza del raggio (se il raggio parte all'interno della regione "voxelizzata"
  - Primo voxel colpito da raggio

DDA-3D: rasterizzazione in 3D [Amanatides&Woo 87]

```
DDA3D(r){
//inizializza X,Y
do{
if(tMaxX < tMaxY){
    tMaxX = tMaxX + tDeltaX;
    X = X + stepX;//stepX ==1 o -1. è il segno di dx
}else
  {
    tMaxY = tMaxY + tDeltaY;
    Y = Y + stepY;
  }
}while( (X,Y) fuori dalla regione ));
}
```

DDA-3D: esempio

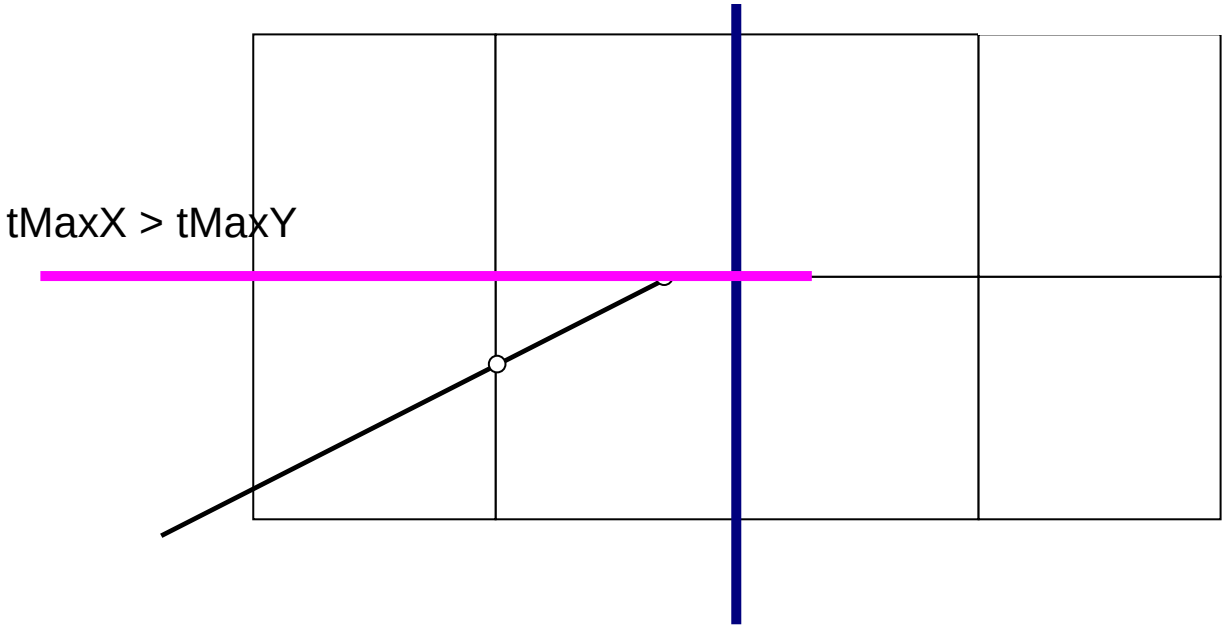
X = 0
Y = 0



tMaxX < tMaxY

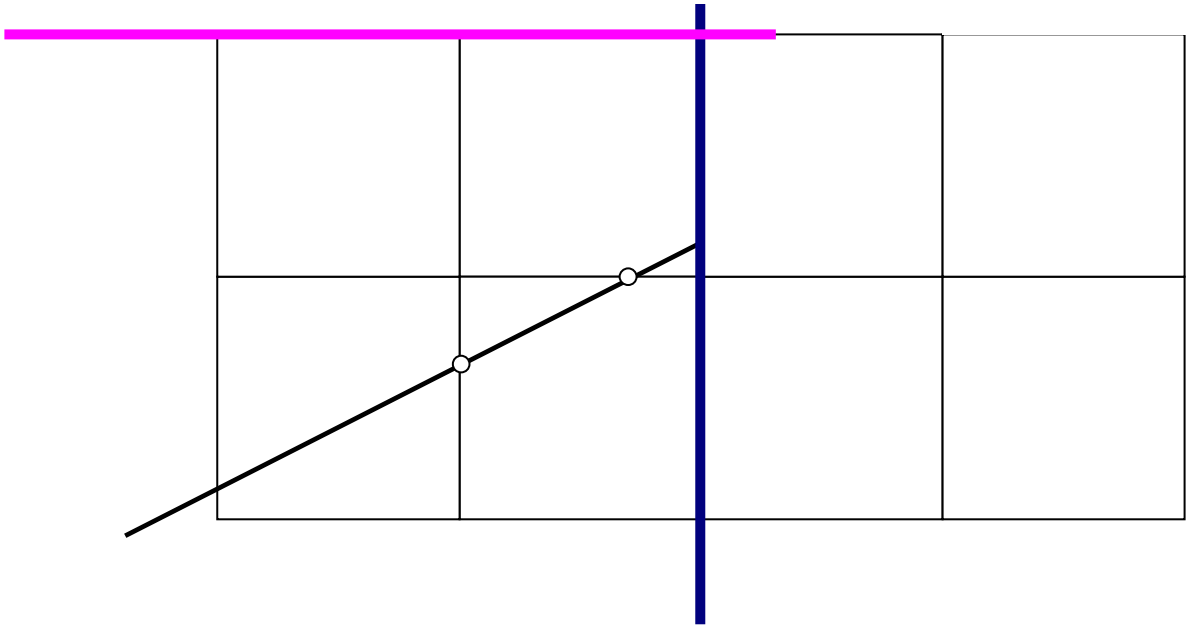
DDA-3D: esempio

X = 1
Y = 0



DDA-3D: esempio

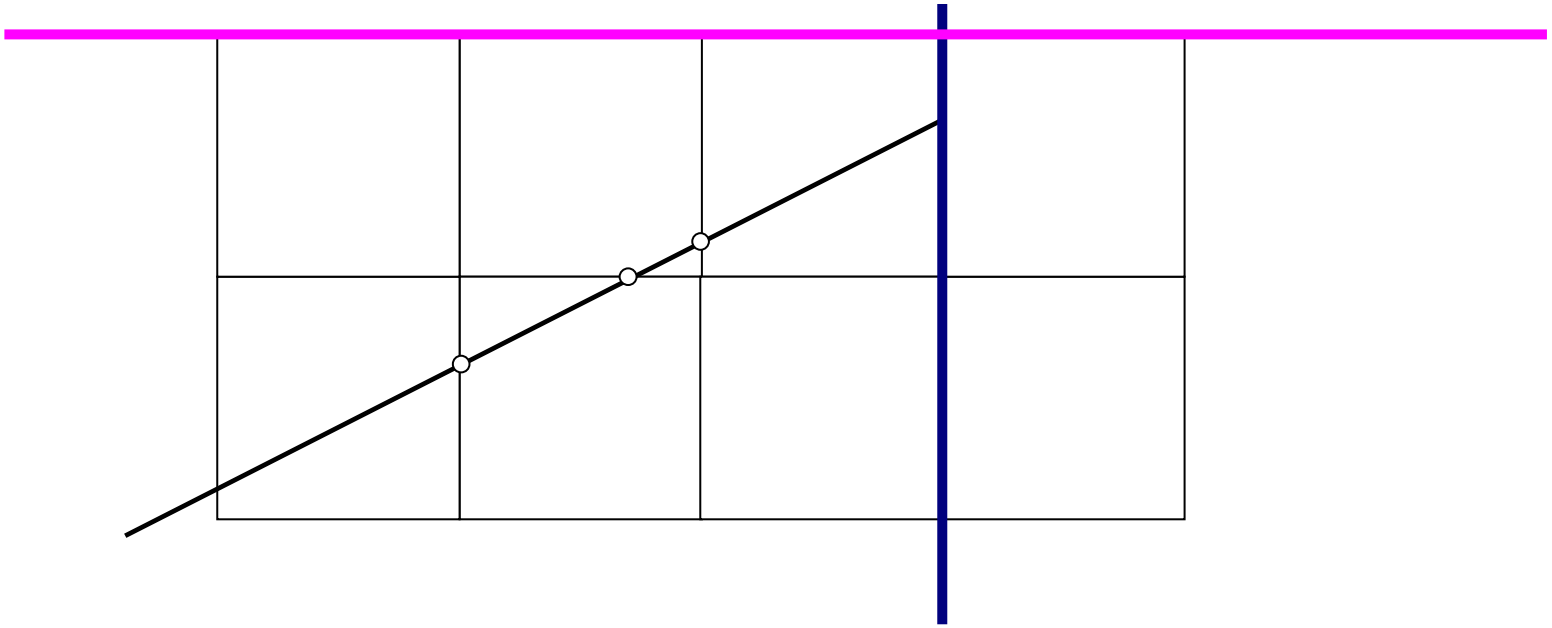
X = 1
Y = 1



tMaxX < tMaxY

DDA-3D: esempio

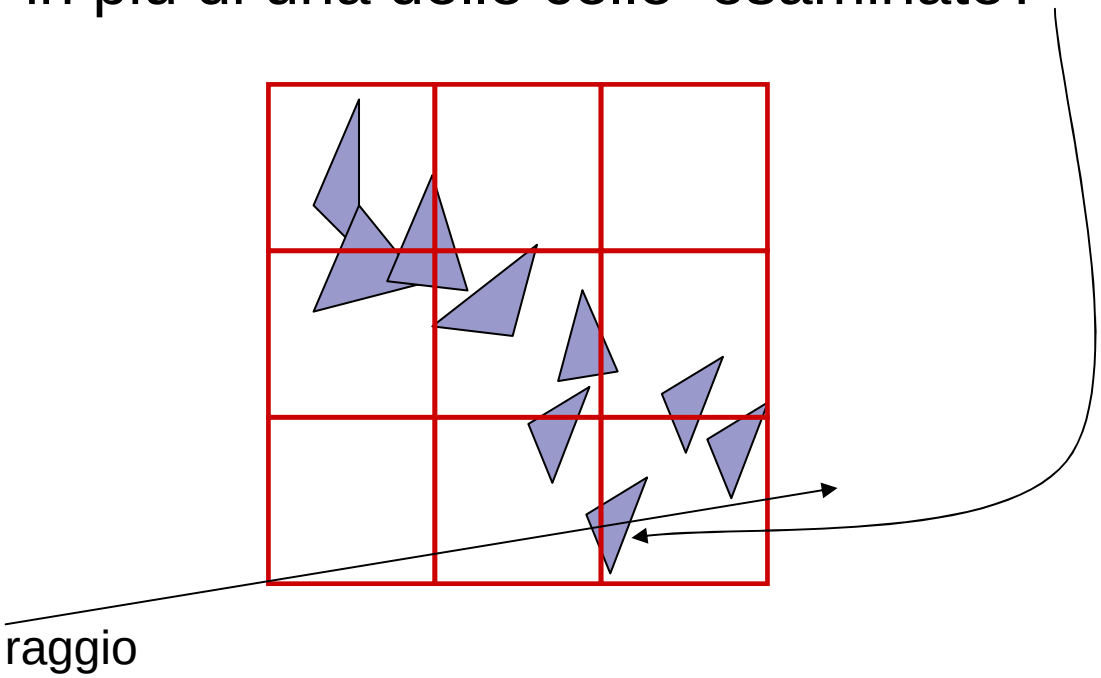
X = 2
Y = 1



tMaxX < tMaxY

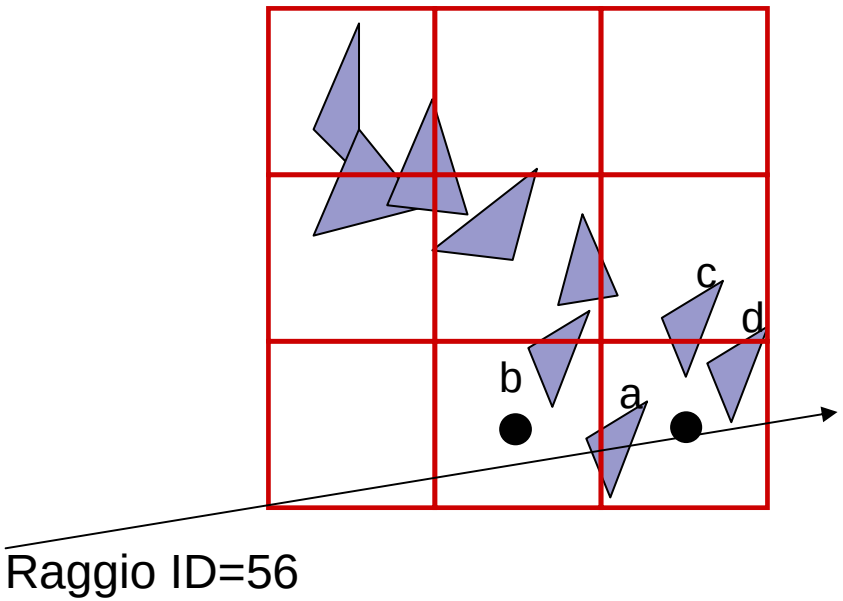
Quanti test di intersezione?

- Usando la griglia regolare abbiamo utilizzato la seguente osservazione
  - Se il raggio non interseca una cella, di sicuro non interseca niente che sia interamente contenuto nella cella
- Quante volte viene testata una primitiva che è presente in più di una delle celle esaminate?



Mailboxing

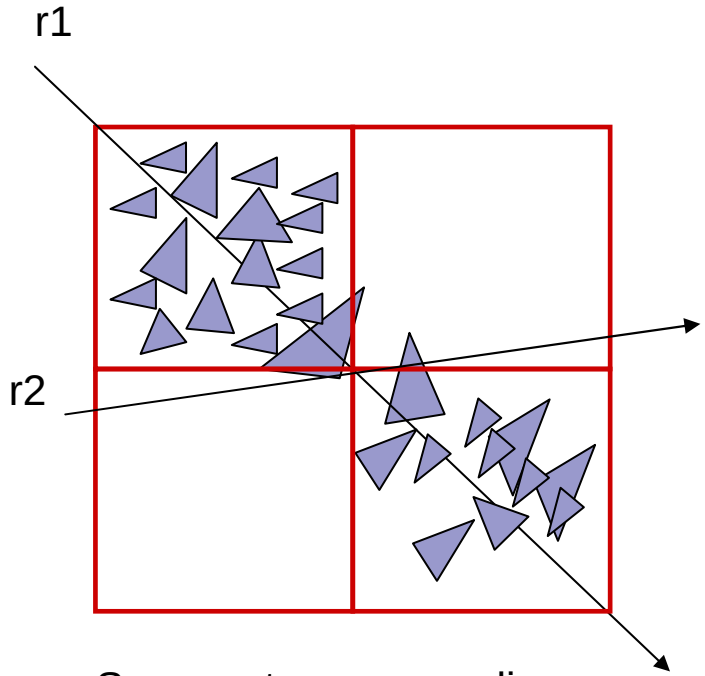
- Ogni raggio ha un suo ID (es: un intero)
- Ad ogni primitiva si associa l'ID dell'ultimo raggio per cui è stato eseguito il test di intersezione



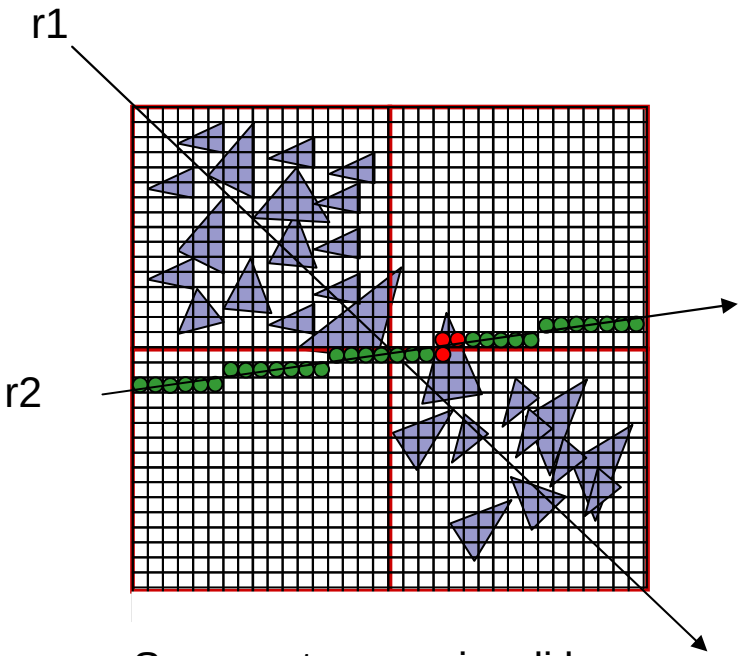
Intersection(raggio,b)
b.ID = 56;
Intersection(raggio,a)
a.ID = 56;

Skip Intersection(raggio,a)
Intersection(raggio,c)
c.ID=56;
Intersection(raggio,d)
d.iD=56;

Quanto devono essere grandi i voxel?



Se sono troppo grandi devo comunque fre molti test di intersezione con le primitive

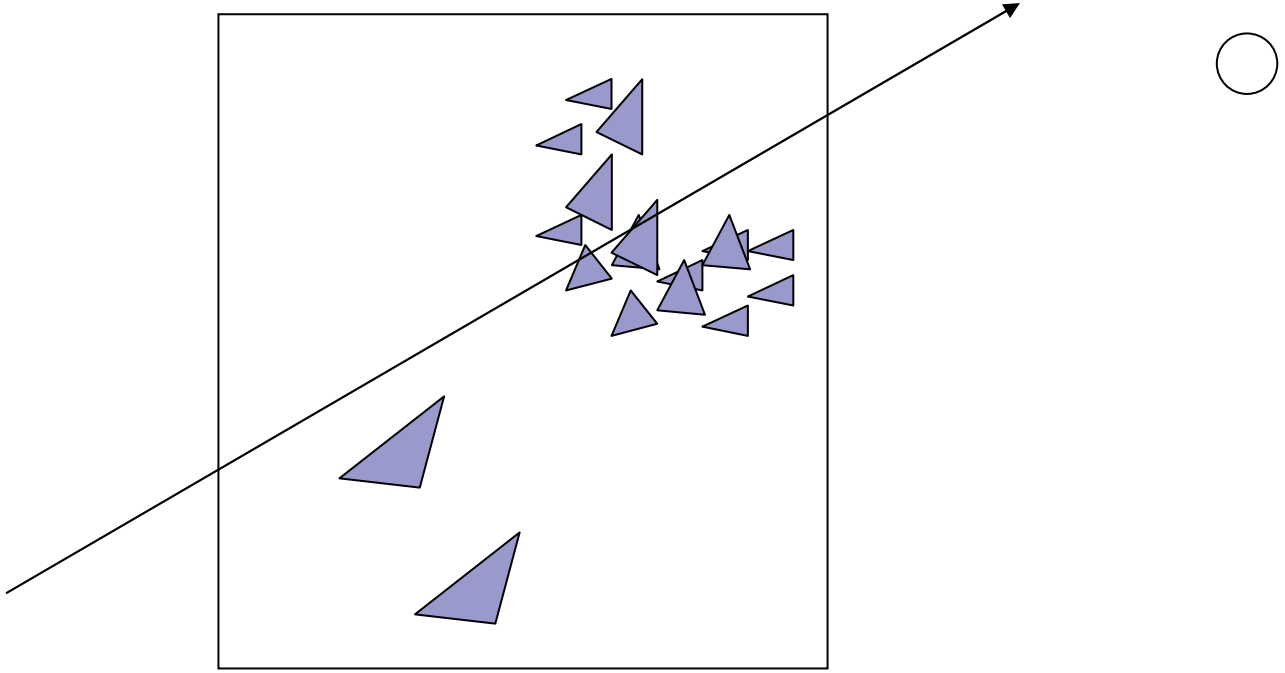


Se sono troppo piccoli la rasterizzazione rallenta

Nelle zone in cui le primitive sono densamente distribuite, converrebbe avere voxel più piccoli
Nelle zone in cui le primitive sono poche, converrebbe avere voxel più grandi

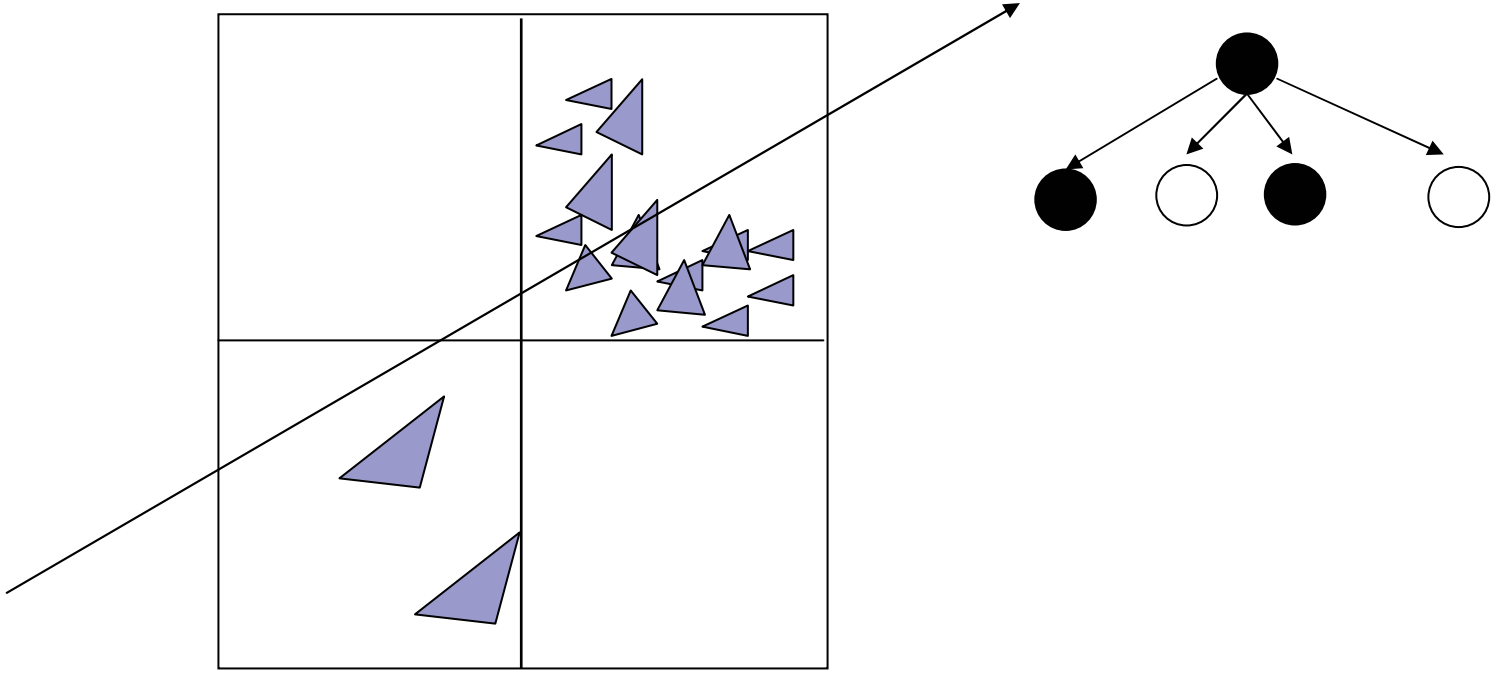
Strutture di indicizzazione spaziale gerarchiche

- Il principio usato per le griglie regolari:
  - Se un raggio non interseca un voxel, di sicuro non interseca niente ivi contenuto
- Applichiamo questo principio ricorsivamente



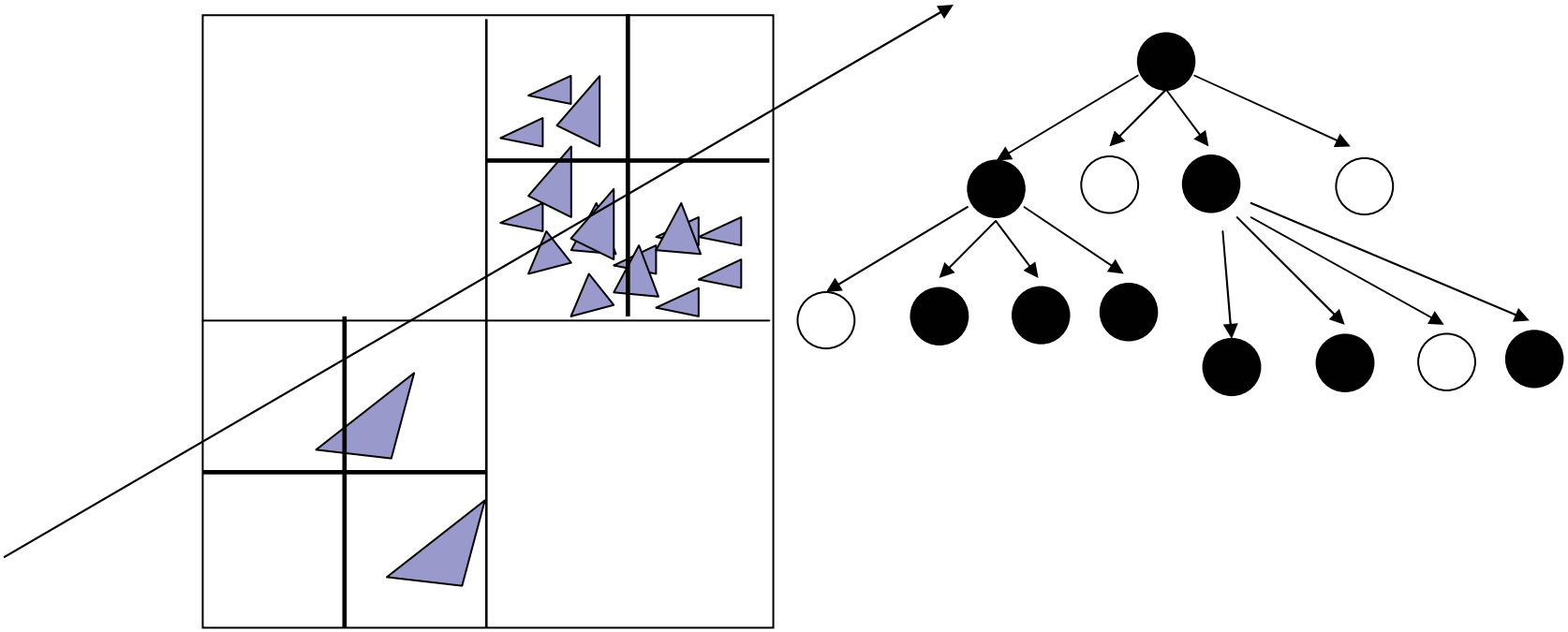
Strutture di indicizzazione spaziale gerarchiche

- Il principio usato per le griglie regolari:
  - Se un raggio non interseca un voxel, di sicuro non interseca niente ivi contenuto
- Applichiamo questo principio ricorsivamente



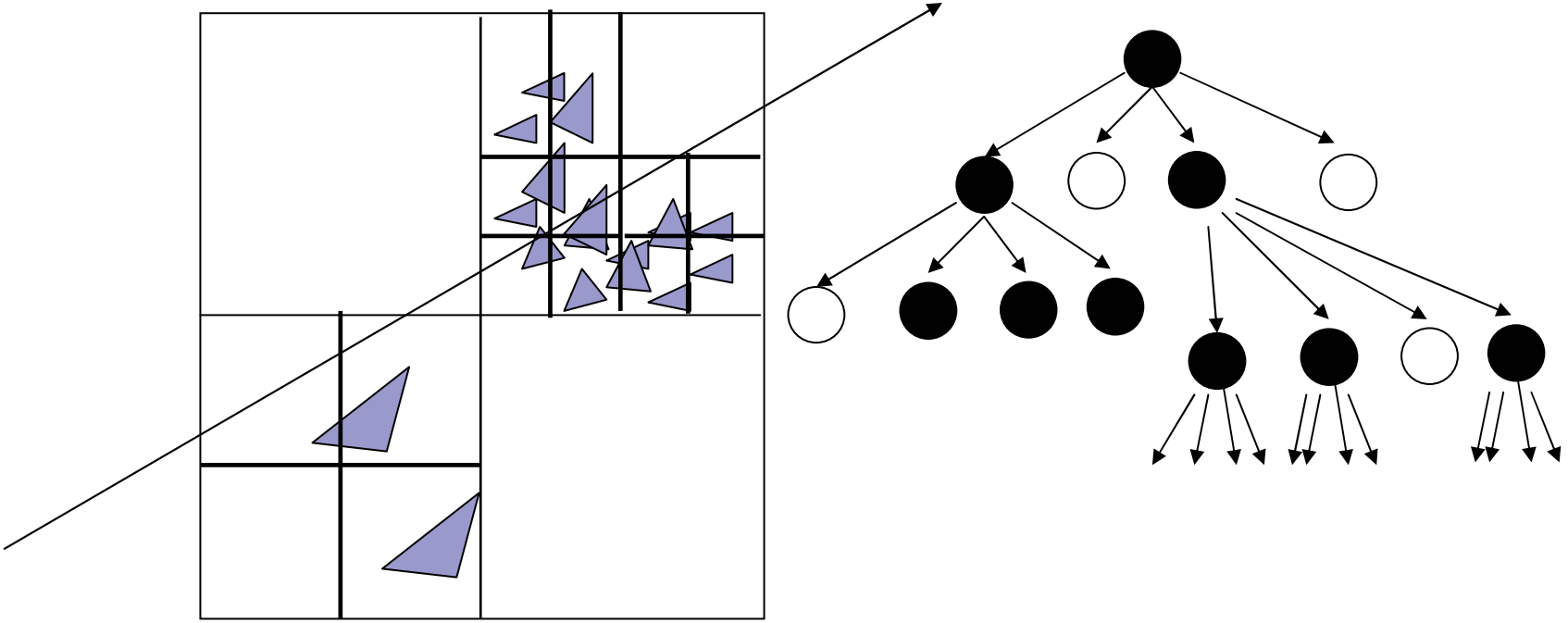
Strutture di indicizzazione spaziale gerarchiche

- Il principio usato per le griglie regolari:
  - Se un raggio non interseca un voxel, di sicuro non interseca niente ivi contenuto
- Applichiamo questo principio ricorsivamente



Strutture di indicizzazione spaziale gerarchiche

- Il principio usato per le griglie regolari:
  - Se un raggio non interseca un voxel, di sicuro non interseca niente ivi contenuto
- Applichiamo questo principio ricorsivamente



Algoritmo

```
TestTree(Ray ray,Node node, float &t, Primitive p){
toVisit = {node}
t = infinite; intersected = NULL;
while( (toVisit != {}) && (t == infinite) ){
n = pop(toVisit);
if( Intersect(ray,n) )
   if(IsLeaf(n)){
       for each primitive p in node
          t' = Intersect(ray,p);
          if(t' < t)
           {
           t = t';
           intersected = p;
           }
    } else
   {
          for n' children of n
                 toVisit = toVisit U n'
   }
}// end while


} // end function
```

Rasterizzazione

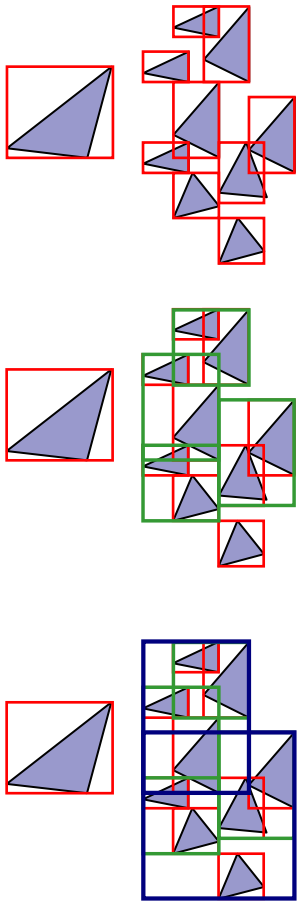
Intersezione raggio primitiva

bool IsLeaf(n){
return n.contained_primitives < *min_prim*
}

Scelta possibile (non unica): se il numero di primitive contenute nel nodo è minore di un valore prefissato *min_prim* allora il nodo è una foglia, cioè la cella corrispondente non viene ulteriormente suddivisa

Strutture di indicizzazione spaziale gerarchiche

- Due tipi di suddivisione:
- Suddivisione delle primitive

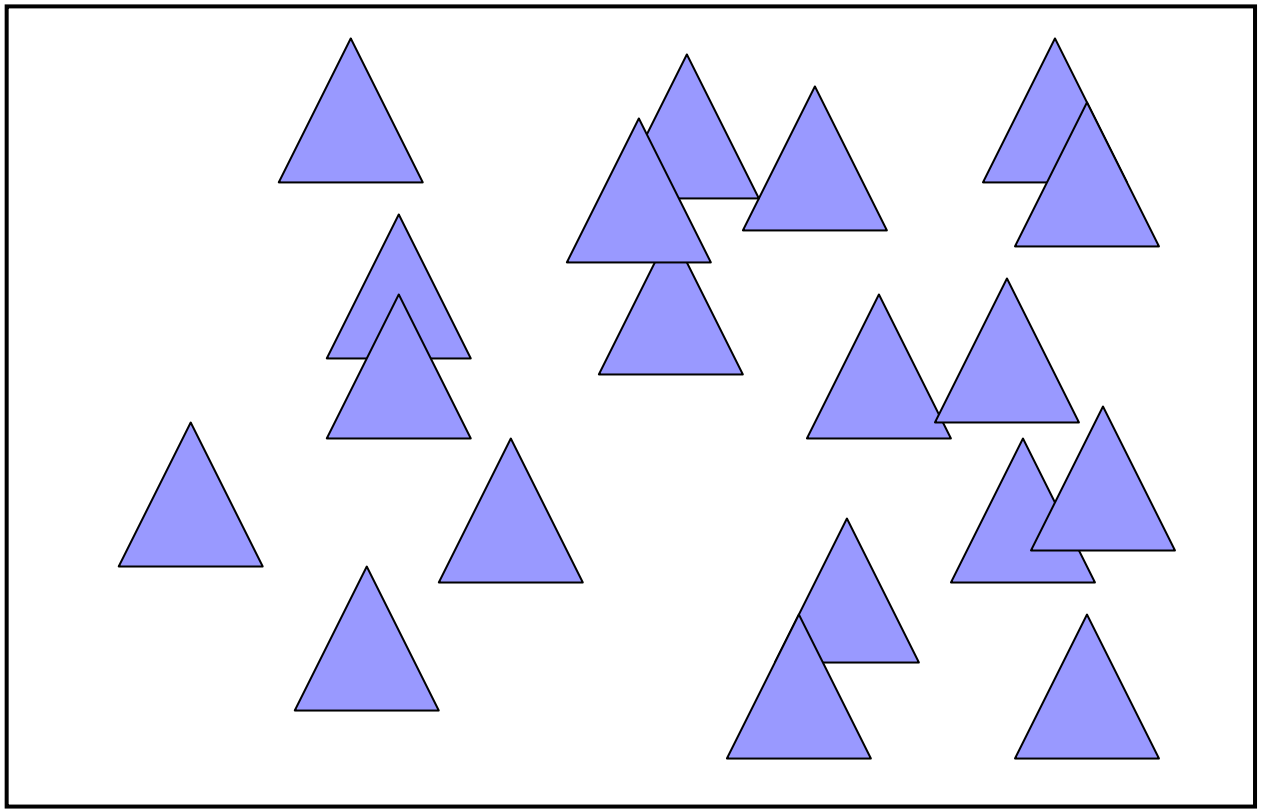


- In genere costruita Bottom-Up
- Ogni primitiva è interamente dentro il nodo
- I BV si intersecano (male, aumenta il numero di intesezioni raggio BV
- I BV sono disposti in maiera irregolare (male, non posso rasterizzare)

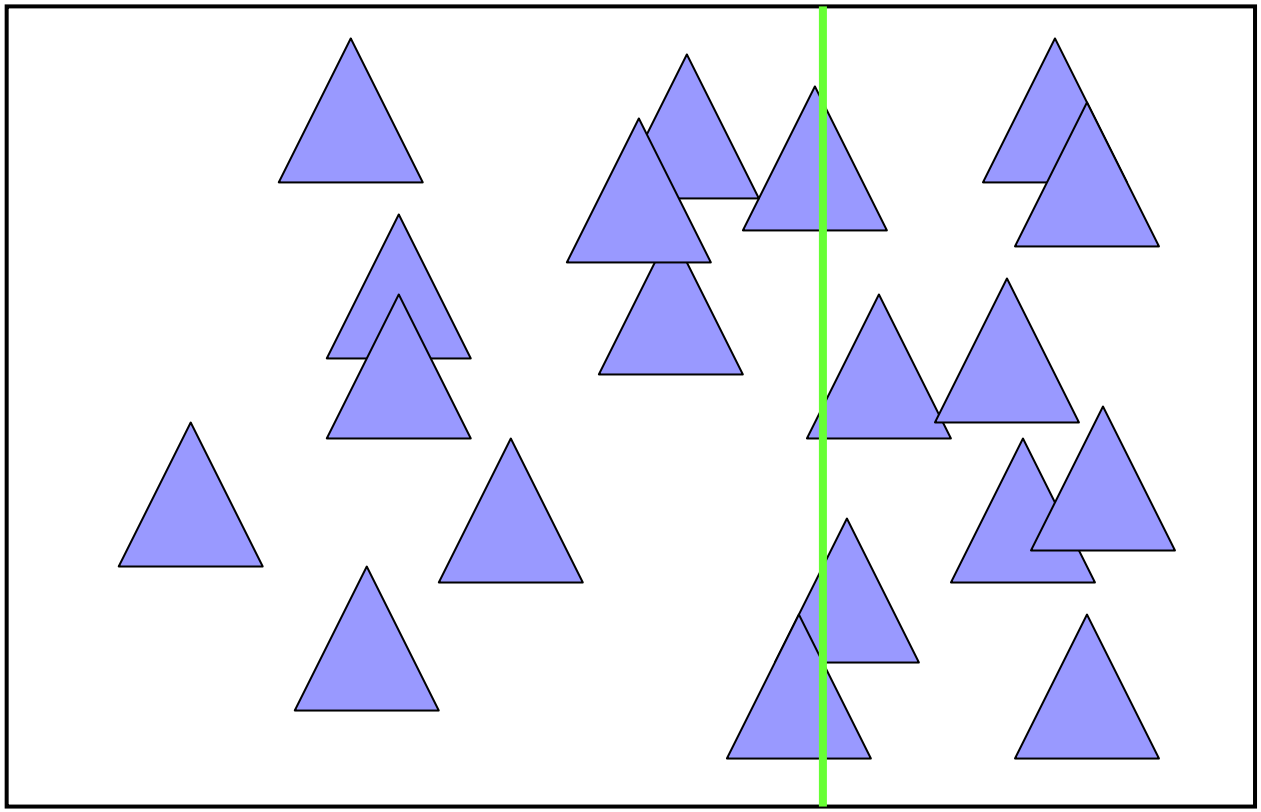
Strutture di indicizzazione spaziale gerarchiche

- Due tipi di suddivisione:
- Suddivisione dello spazio (come nell'esempio):
  - In genere costruita Top-Down
  - Ogni primitiva può essere interamente dentro il nodo o no (nell'esempio no)
  - Le celle non si intersecano
  - Le celle sono disposte in maniera regolare o irregolare (nell'esempio in maniera regolare)
- Limitiamoci a quella più adatta per il il Ray-Tracing

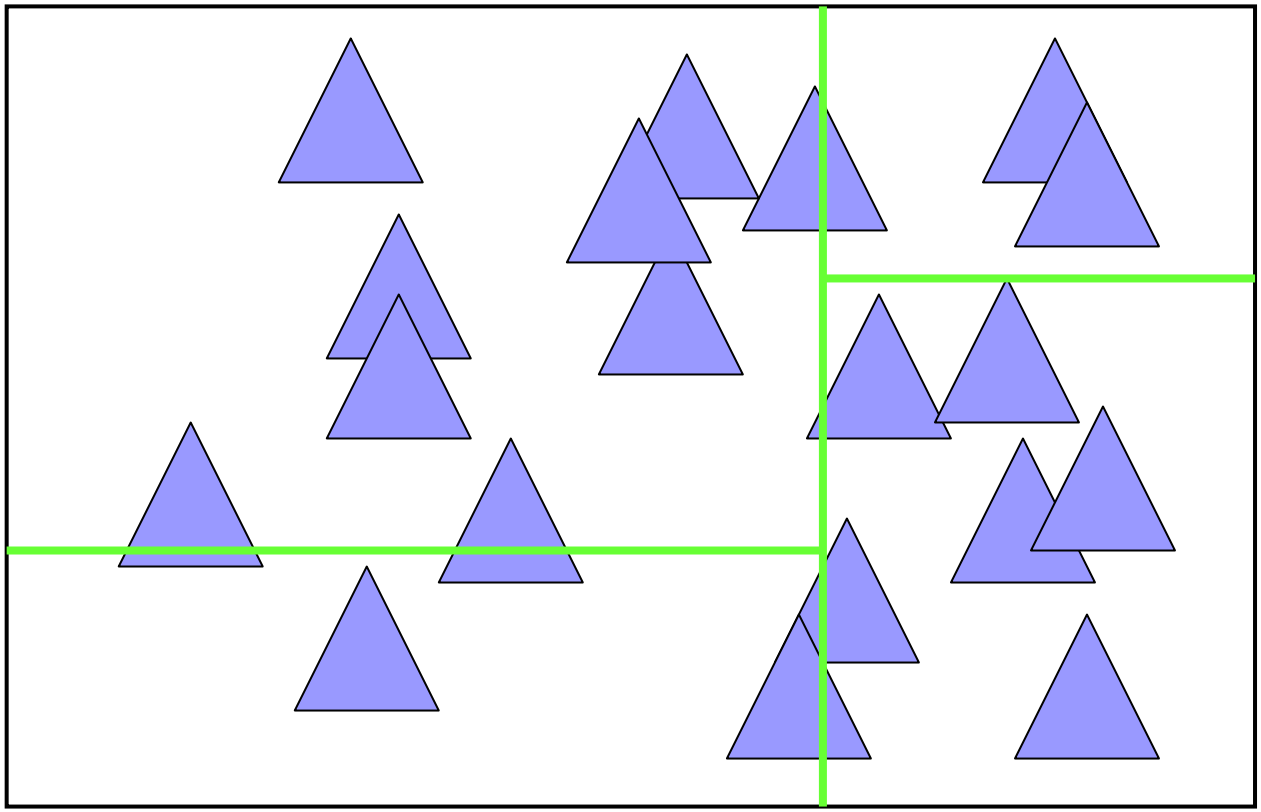
kD-Trees



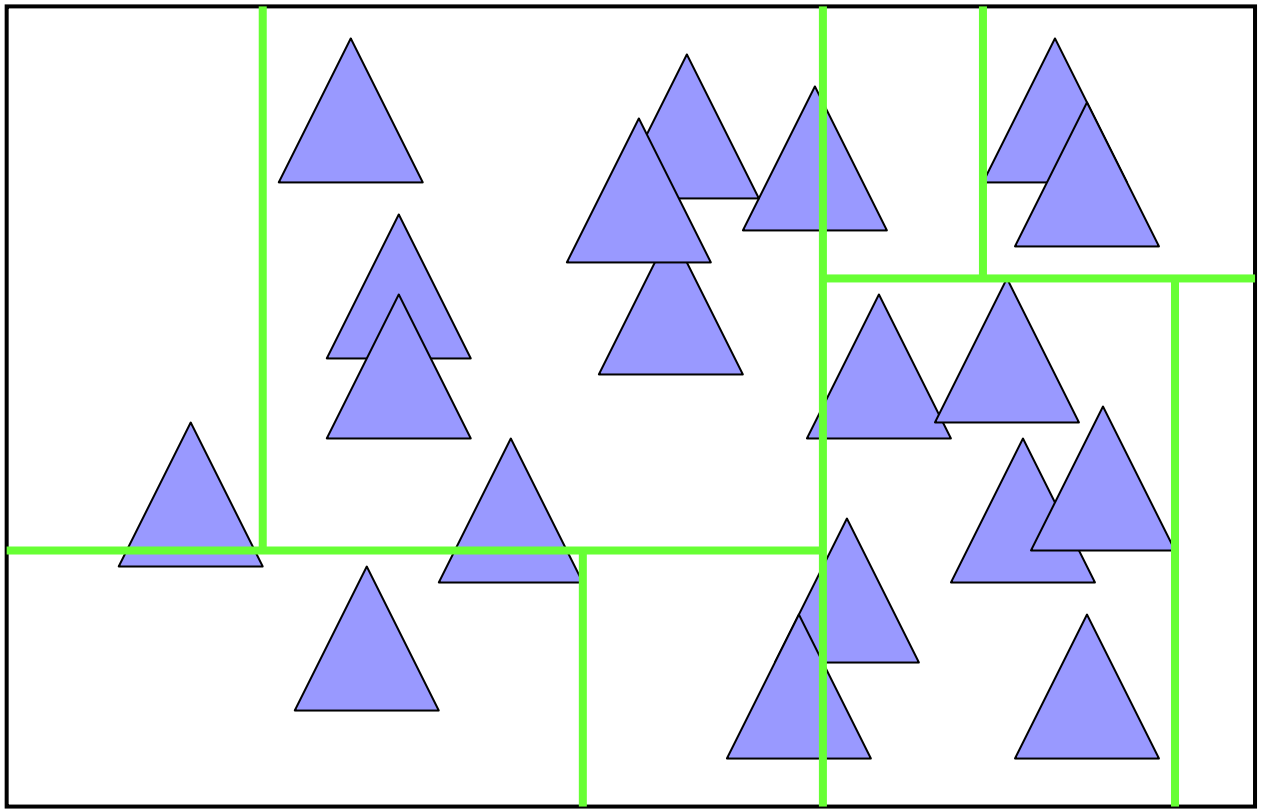
kD-Trees



kD-Trees



kD-Trees



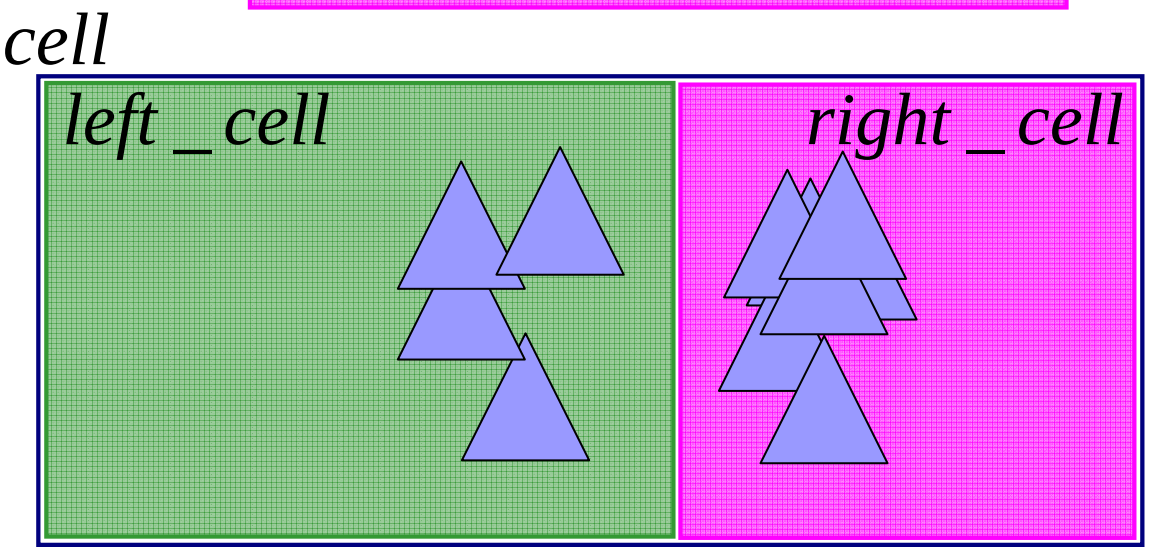
Costruire un kD-tree

- Dati:
  - axis-aligned bounding box (“cell”)
  - lista di primitive geometriche (triangoli)
- Operazioni base
  - Prendi un piano ortogonale a un asse e dividi la cella in due parti **(in che punto?)**
  - Distribuire le primitive nei due insiemi risultanti
  - Ricorsione
  - Criterio di terminazione **(che criterio?)**

Costruire un kD-tree efficiente

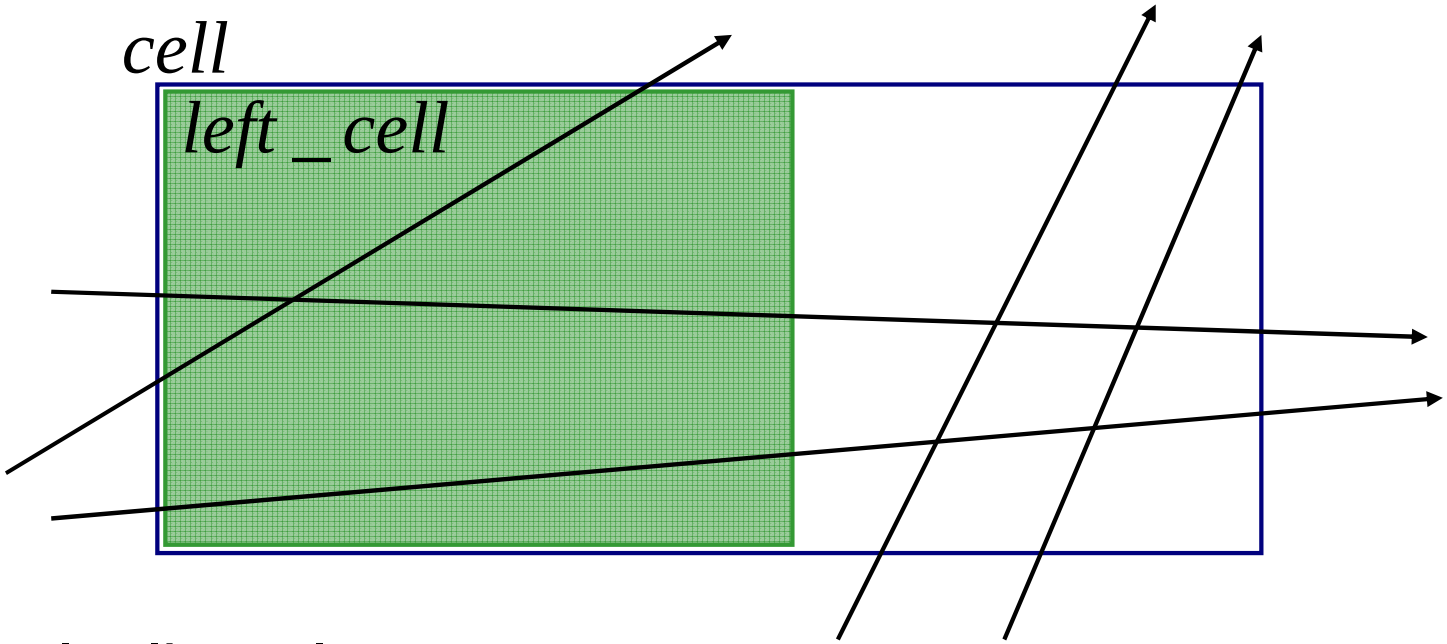
- In che punto dividere la cella?
  - Nel punto che minimizza il **costo**
- Quanto è il costo?

$$Cost(cell) = Cost\_traversal +$$
$$\text{Prob}(left\_cell \mid cell)\, Cost(Left) +$$
$$\text{Prob}(right\_cell \mid cell)\, Cost(Right)$$



$\text{Prob}(left\_cell \mid cell)\, Cost(Left)$

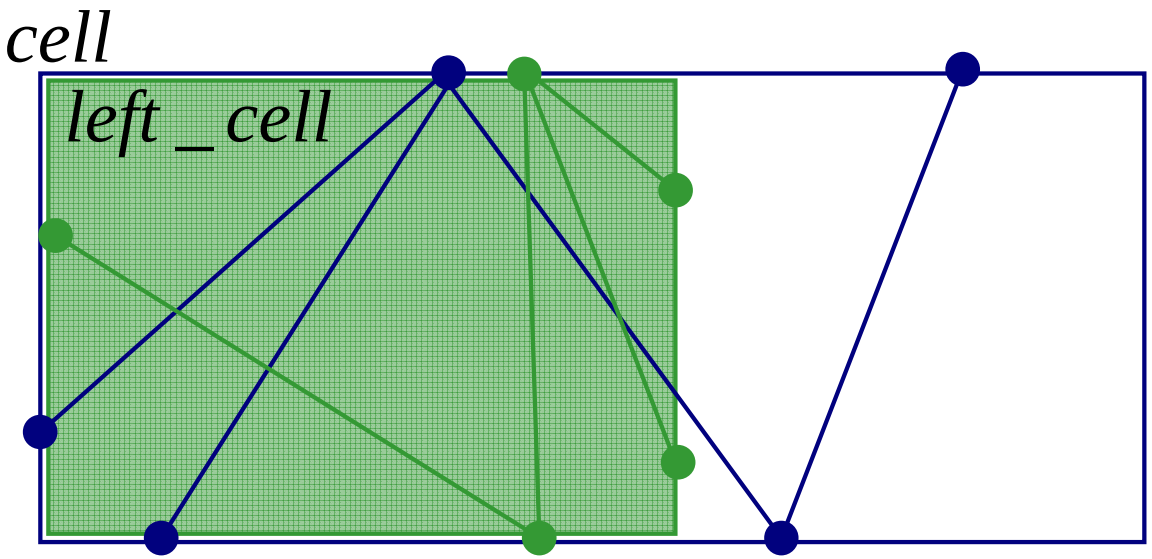
- Sapendo che il raggio interseca la cella *cell*, qual'è la probabilità che intersechi la cella *left_cell* ??



- Calcoliamola….

$\text{Prob}(left\_cell \mid cell)$

$$\text{Prob}[cell \mid left\_cell] = \frac{\#\, raggi\ che\ intersecano\ left\_cell}{\#\, raggi\ che\ intersecano\quad cell}$$

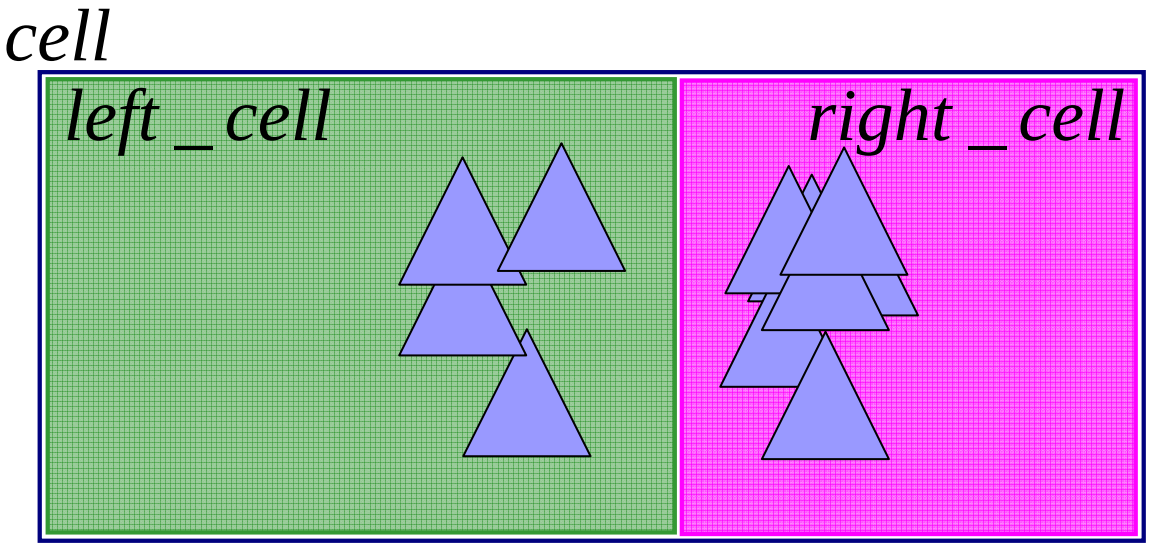


Ogni raggio che interseca una cella corrisponde a una coppia di punti sulla sua superficie. Contiamo le coppie di punti sulla superficie delle celle

$$\text{Prob}[cell \mid left\_cell] = \frac{\int_{\sigma(left\_cell)} \left(\int_{\sigma(left\_cell)} da \right) da}{\int_{\sigma(cell)} \left(\int_{\sigma(cell)} da \right) da} = \frac{Area(left\_cell)^2}{Area(cell)^2} = \frac{Area(left\_cell)}{Area(cell)}$$

$cost(left\_cell)$

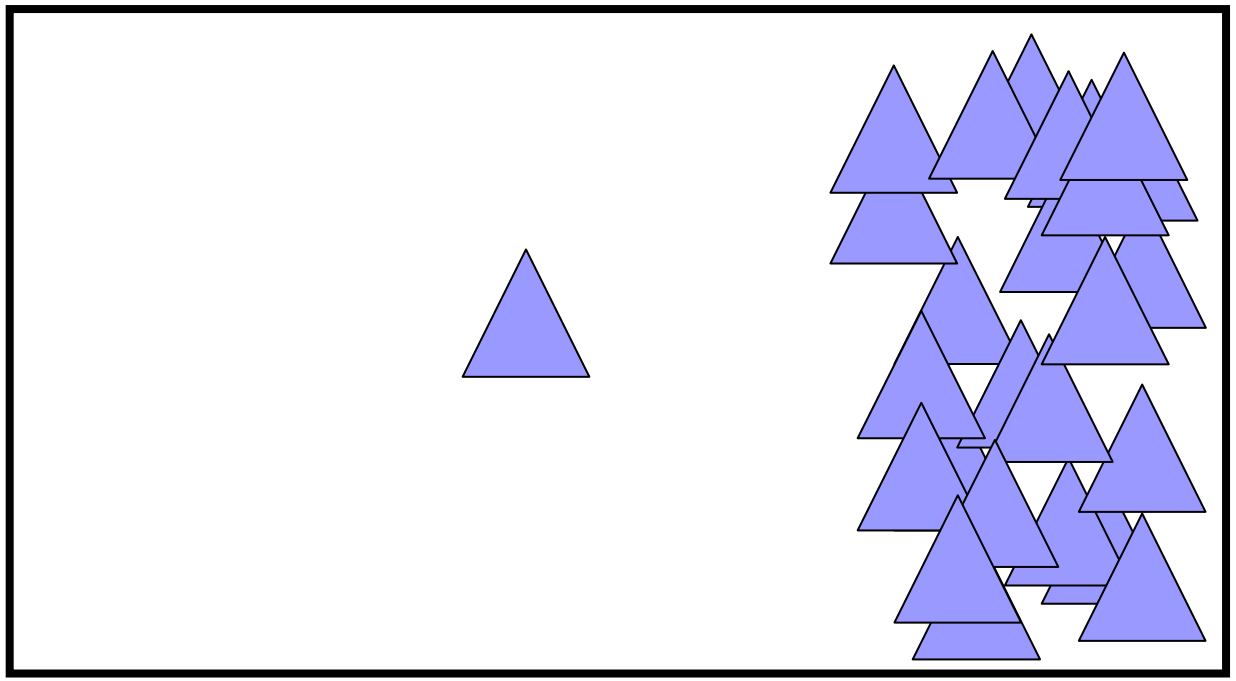
- Sapendo che il raggio interseca la cella *left_cell*, qual'è il costo di testare l'intersezione con i triangoli?
- Si approssima con il numero di triangoli che toccano la cella



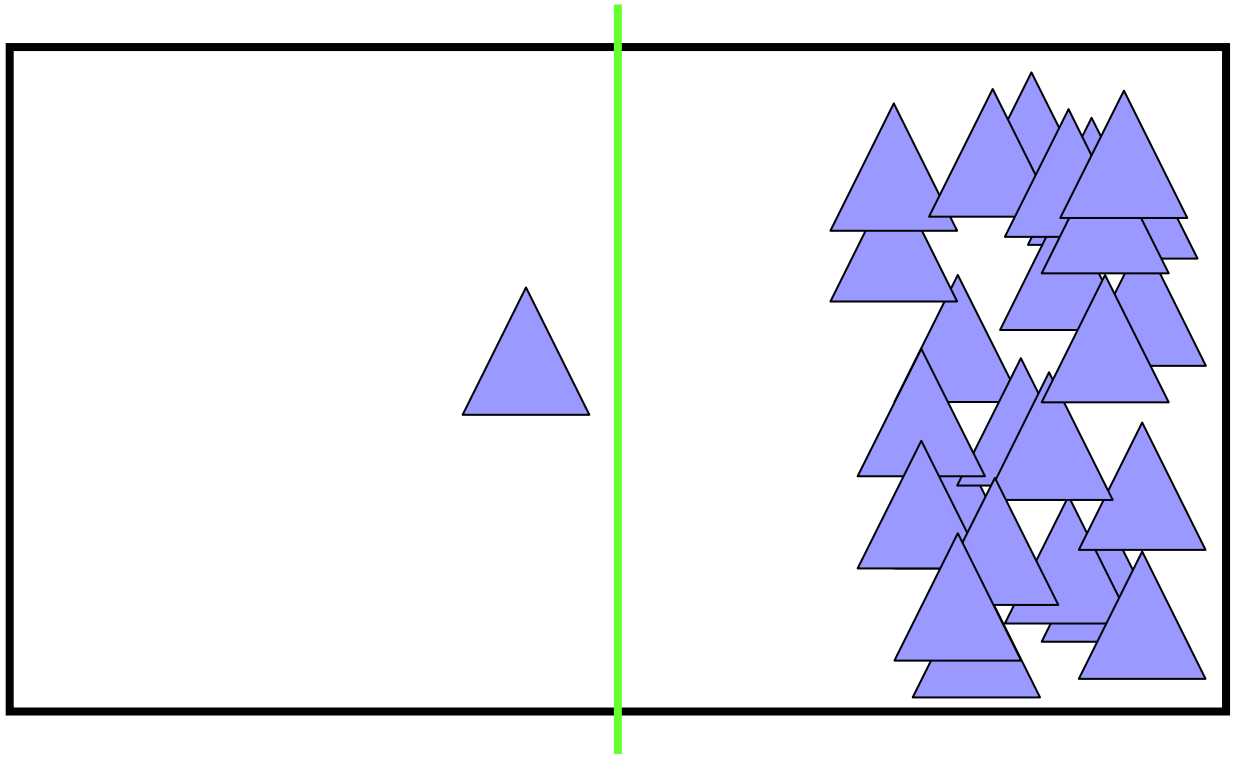
$$Cost(left\_cell) = 4$$

Esempio

- Come si suddivide la cella qui sotto?

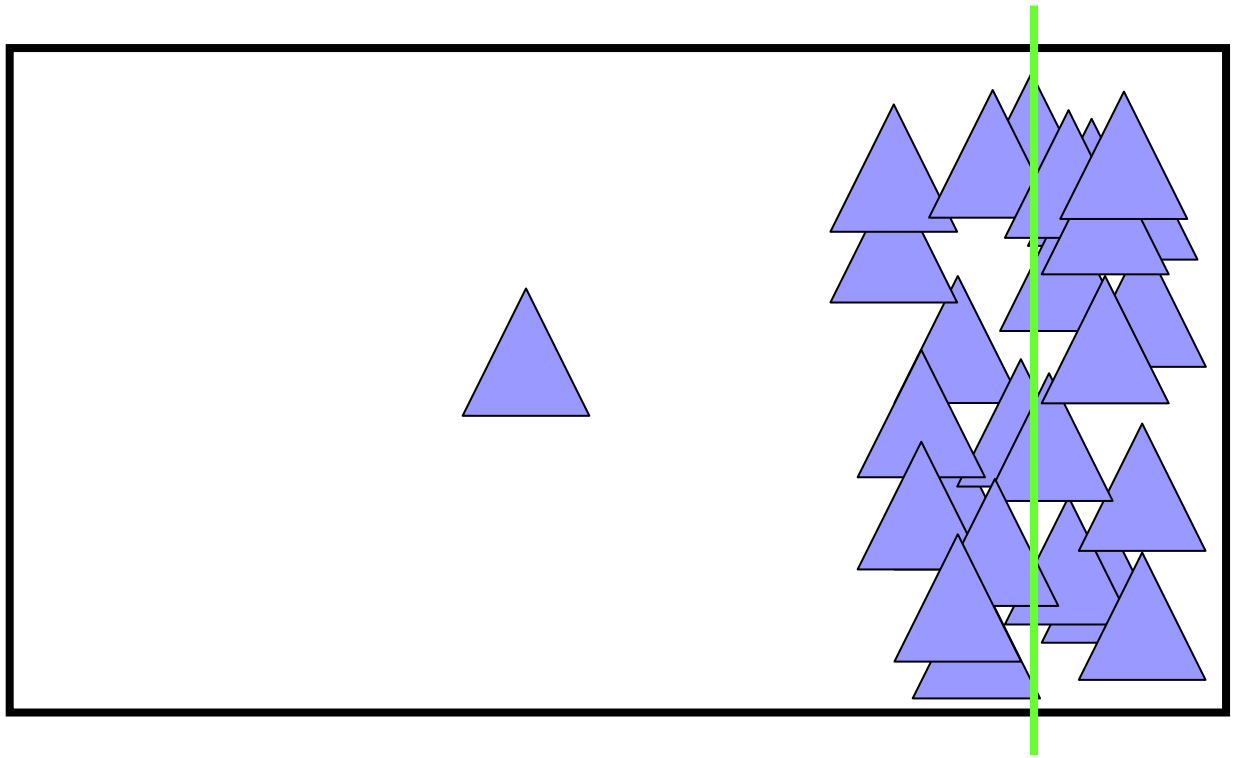


A metà



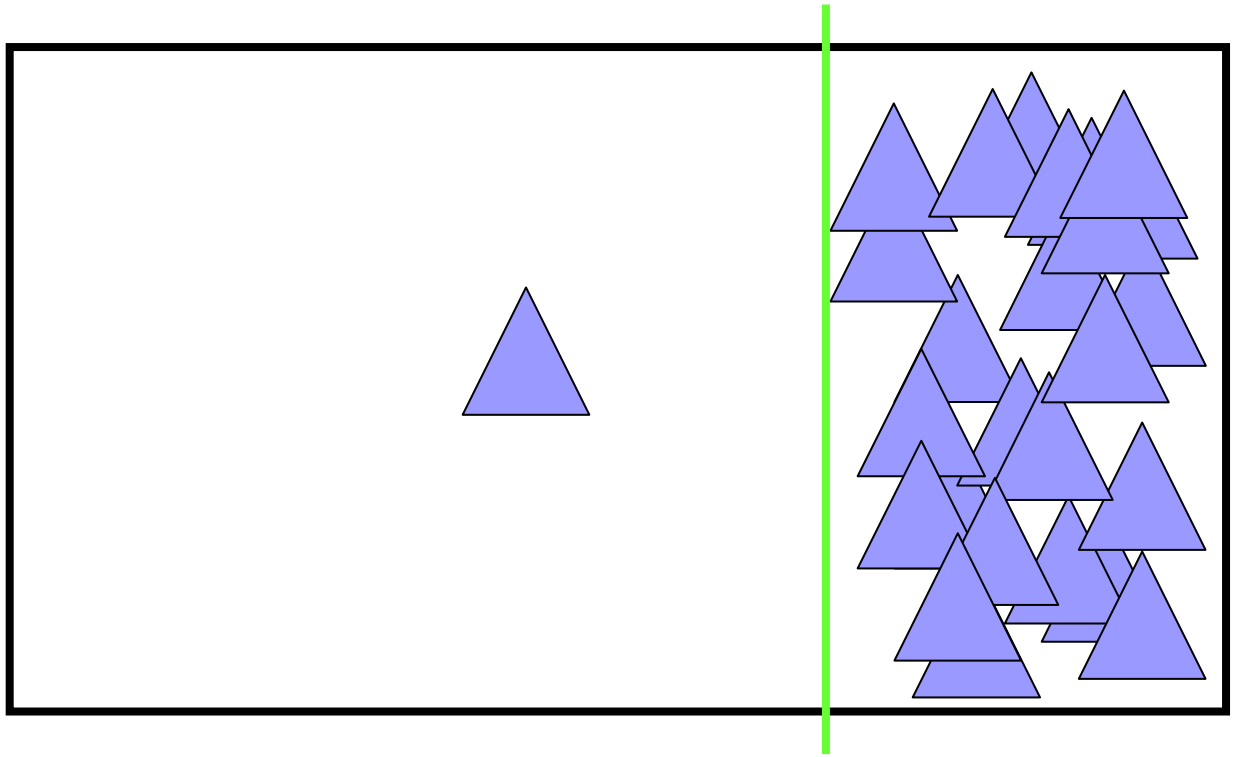
- Non tiene conto delle probabilità
- Non tiene conto dei costi

Nel punto mediano



- Rende uguali i costi di *left_cell* e *right_cell*
- Non tiene conto delle probabilità

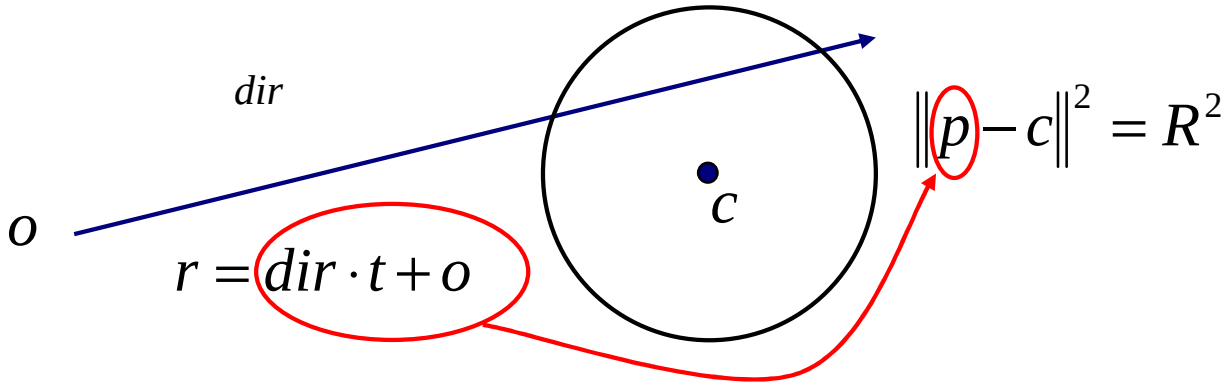
Ottimizzando il costo



- Separa bene spazio vuoto
- Distribuisce bene la complessità

Intersezioni geometriche: raggio-sfera

- Algebricamente:



$$\|dir \cdot t + o - c\|^2 = R^2$$

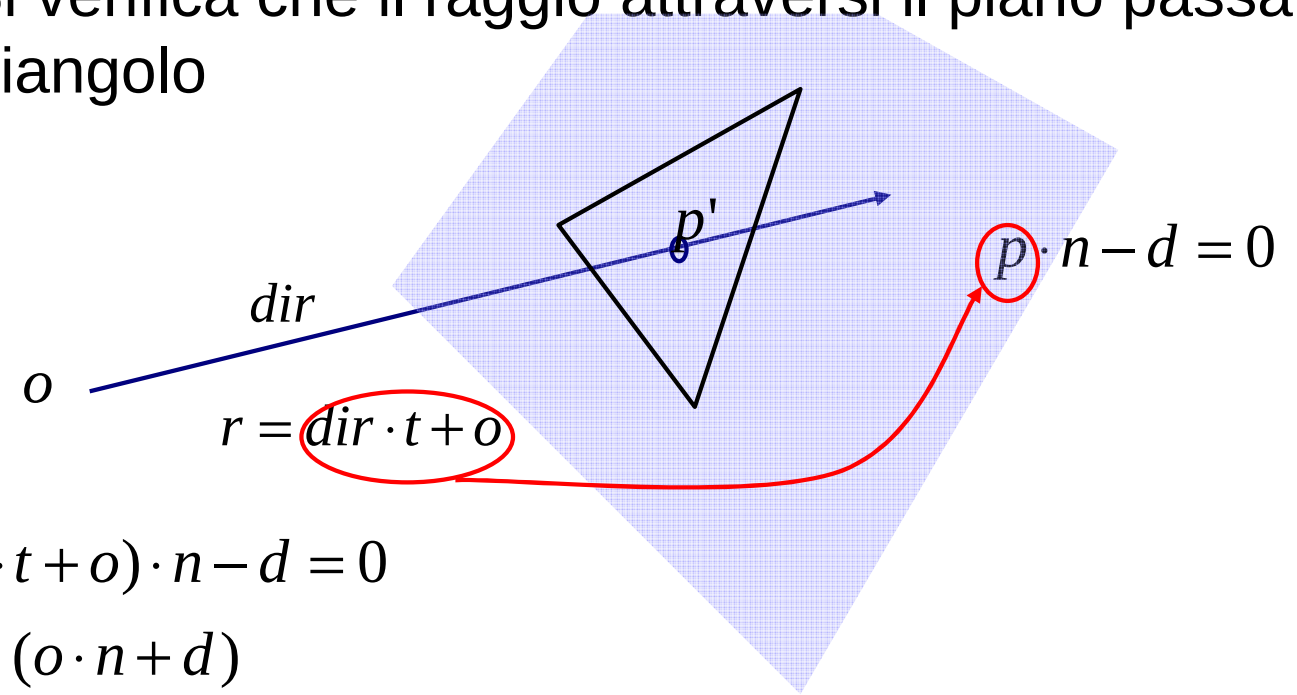
$$t = -B \pm \sqrt{B^2 - 4AC}$$

$$dove:$$

$$B = 2\ dir\,(o - c)\ \ ,\ \ A = \|\,dir\,\|^2\ ,\ C = \|\,dir - c\|^2$$

Intersezioni geometriche: raggio-triangolo

1. Si verifica che il raggio attraversi il piano passante per il triangolo



$$(dir \cdot t + o) \cdot n - d = 0$$

$$t = -\frac{(o \cdot n + d)}{dir \cdot n}$$

$$p' = dir \cdot \left(-\frac{(o \cdot n + d)}{dir \cdot n} \right) + o \qquad dir \cdot n = 0 \Rightarrow raggio\ \ parallelo\ al\ \ piano$$

Rimane da determinare se il punto p' è interno al triangolo
Basta calcolarne le coordinate baricentriche..