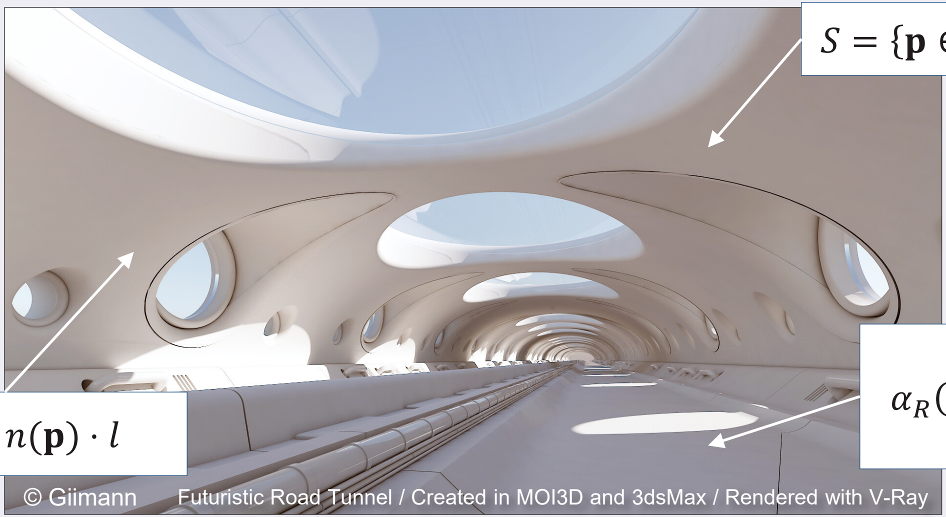
# Computer Graphics

From mathematics …

$$S = \{\mathbf{p} \in \mathbf{R}^3, f(\mathbf{p}) = 0\}$$



© Giimann Futuristic Road Tunnel / Created in MOI3D and 3dsMax / Rendered with V-Ray

$$d(\mathbf{p}) = n(\mathbf{p}) \cdot l$$

$$\alpha_R(\mathbf{p}) \approx \frac{1}{n} \sum_{i=0}^{n} \delta_i$$

… to the screen

E. Galin
Université Lyon 1

# Computer Graphics

Mathematics
Modeling
Color and Texturing
Shading
Realistic Rendering
Acceleration
Animation
**Ray Tracing**

# Computer Graphics

## Architecture

# Architecture

## Structures et notations

Camera $E$, générant des rayons $\Delta$ pour chaque pixel $\mathbf{p}_{ij}$

Chaque objet $S_k$ de la scène $S$ possède un indice $k$ vers un matériau $M_k$

Lumières $L$

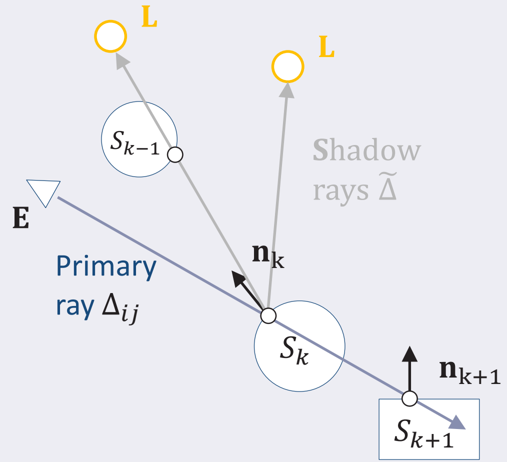
**Render** $(E)$
For all pixels $\mathbf{p}_{ij}$
Create ray $\Delta_{ij}$
$c_{ij} \leftarrow$ **Shade** $(\Delta_{ij})$

**Shade** $(\Delta_{ij})$
$\mathrm{X} \leftarrow$ **Intersection** $(S, \Delta_{ij})$
$c_{ij} \leftarrow$ **Lighting** $(\Delta_{ij}, \mathrm{X})$

**Lighting** $(\Delta_{ij}, \mathrm{X})$
Compute $c_{ij}$ from material (in $\mathrm{X}$)
Compute **Intersection** $(S, \widetilde{\Delta})$

**Intersection** $(S, \Delta_{ij})$
For all objects $S_k \in S$
Compute $X_k \leftarrow$ **IntersectionType** $(S_k, \Delta_{ij})$
Select closest intersection

**IntersectionType** $(S_k, \Delta_{ij})$
Solve $t \leftarrow S_k \cap \Delta_{ij}$
Compute data for hit $X_k = \{t, \mathbf{n}, M_k\}$




eric.galin@liris.cnrs.fr
http://liris.cnrs.fr/~egalin

# Computer Graphics

## Camera

# Camera rays



## Objectifs

Projeter un point $\mathbf{p} \in \mathbf{R}^3$ de l'espace sur l'écran en un pixel $\mathbf{q} \in [0, w-1] \times [0, h-1] \subset \mathbf{N}^2$

Définir l'équation d'un rayon $\Delta$ à partir d'un pixel $\mathbf{q}$

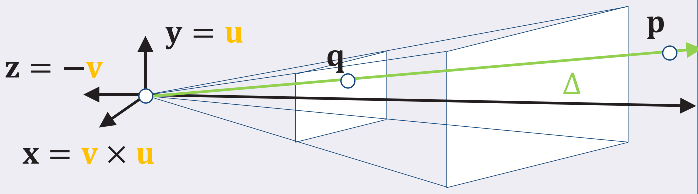
## Caractérisation

Position $\mathbf{e}$, vecteur de vue $\mathbf{v}$, vecteur haut $\mathbf{u}$

Angle d'ouverture horizontal $\alpha$, demi angle $\beta = \alpha/2$

Aspect ratio $r = w/h$

Demi hauteur $\tilde{h} = h/2$ et longueur $\tilde{w} = w/2$




Rayon $\Delta(\mathbf{e}, \mathbf{d})$ depuis un pixel $\mathbf{q}$

$$\mathbf{d} = \tan\beta \frac{(\mathbf{q}_x - \tilde{w})}{\tilde{w}} \mathbf{x} + \frac{1}{r} \tan\beta \frac{(\tilde{h} - \mathbf{q}_y)}{\tilde{h}} \mathbf{y} + \mathbf{v}$$

Coordonnées unitaires dans l'écran

Repère local

$\mathbf{x} = \mathbf{v} \times \mathbf{u}$

$\mathbf{y} = \mathbf{u}$

$\mathbf{z} = -\mathbf{v}$


Université Claude Bernard Lyon 1

eric.galin@liris.cnrs.fr

http://liris.cnrs.fr/~egalin

# Computer Graphics

## Intersections

# Sphère



## Intersection

Sphère $S$ de centre $\mathbf{c}$ et rayon $r$
Equation paramétrique du rayon $\Delta$

$$S = \{\mathbf{p} \in \mathbf{R}^3, f(\mathbf{p}) = (\mathbf{p} - \mathbf{c})^2 - r^2 = 0\}$$

$$\Delta = \{\mathbf{p}\,(t) = \delta(t) = \mathbf{o} + \mathbf{d}t, t \in \mathbf{R}^+\}$$

## Solution analytique

Résoudre l'équation de second degré

$$f \circ \delta(t) = \mathbf{d}^2 t^2 + 2(\mathbf{o} - \mathbf{c}) \cdot \mathbf{d}\, t + (\mathbf{o} - \mathbf{c})^2 - r^2 = 0$$

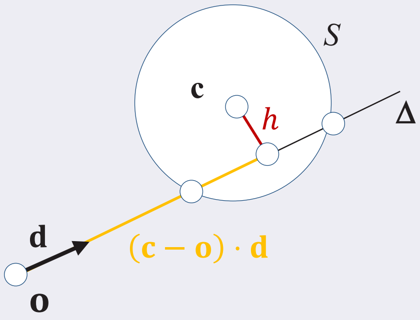
Optimisation : rayon de direction normalisée $|\mathbf{d}| = 1$

Méthode géométrique plus efficace
Calcul de $h^2 = \mathrm{d}(\mathbf{S}, \Delta)^2$ et comparaison à $r^2$

$$h^2 = (\mathbf{c} - \mathbf{o})^2 - \left((\mathbf{c} - \mathbf{o}) \cdot \mathbf{d}\right)^2$$

Les racines se déduisent ensuite

$$t = (\mathbf{c} - \mathbf{o}) \cdot \mathbf{d} \pm \sqrt{r^2 - h^2}$$




Université Claude Bernard Lyon 1
eric.galin@liris.cnrs.fr
http://liris.cnrs.fr/~egalin