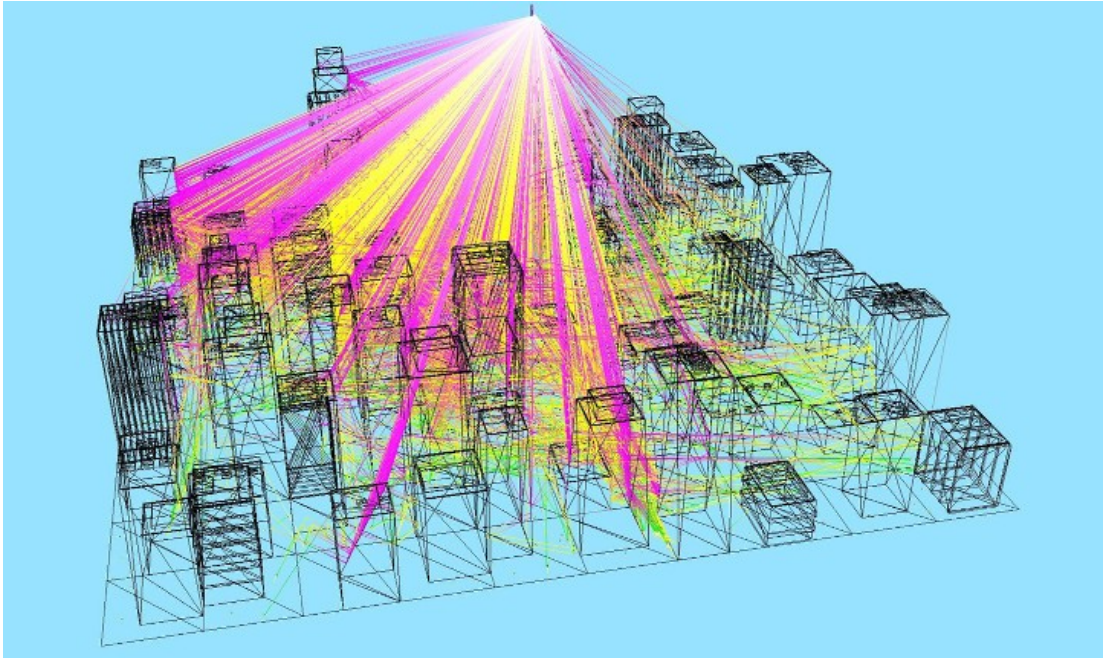
Tema 2

Lanzamiento de rayos



Repaso

- Antes de correr hay que aprender (¡repasar!) como caminar

- Conceptos de:
  - Álgebra lineal
  - Geometría

Vectores

- Es un objeto geométrico que tiene magnitud y dirección

$$\mathbf{v} = (v_x, v_y, v_z)^T$$

- Utilizamos sus tres coordenadas *x, y, z* para denotarlo
  - Un vector siempre “sale” desde el origen
  - No confundir con “rayo”

- La norma es la magnitud del vector y se obtiene:

$$\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

Vectores: suma y resta

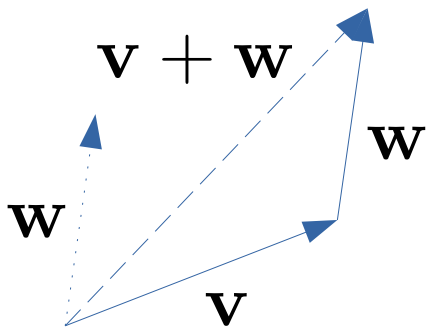
- La suma y la resta de dos vectores se hace sumando o restando cada componente

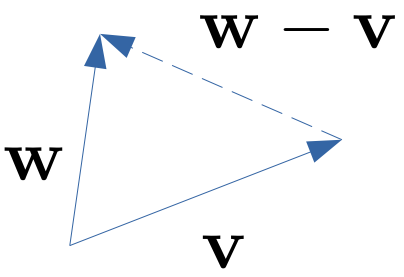
$$\mathbf{a} = \mathbf{v} + \mathbf{w}$$

$$a_x = v_x + w_x$$

$$a_y = v_y + w_y$$

$$a_z = v_z + w_z$$





Vectores: producto con escalar

- El producto de un vector y un escalar se realiza multiplicando cada componente del vector por el escalar

$$a\mathbf{v} = (av_x, av_y, av_z)^T$$

- Se incluye a la división por escalar
  - Multiplica por el recíproco del divisor

$$\frac{\mathbf{v}}{a} = \frac{1}{a}\mathbf{v}$$

Vectores: vector unitario

- Es un vector cuya magnitud es igual a 1
- También llamado *vector normalizado*

- Normalizar un vector (dividir por la norma):

$$\mathbf{v_n} = \frac{\mathbf{v}}{\|\mathbf{v}\|}$$

Vectores: producto punto

- El producto punto se realiza sumando el resultado de los productos componente a componente

$$\mathbf{v} \cdot \mathbf{w} = v_x w_x + v_y w_y + v_z w_z$$

- Relaciona también con el ángulo entre dos vectores:

$$\mathbf{v} \cdot \mathbf{w} = \|\mathbf{v}\| \|\mathbf{w}\| \cos\theta$$

- Si los vectores son unitarios entonces el producto punto es el coseno del ángulo entre ambos vectores

Vectores: propiedades producto punto

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{v} \cdot \mathbf{u}$$

$$s\mathbf{u} \cdot \mathbf{v} = \mathbf{u} \cdot s\mathbf{v} = s(\mathbf{u} \cdot \mathbf{v})$$

$$\mathbf{u} \cdot (\mathbf{v} + \mathbf{w}) = (\mathbf{u} \cdot \mathbf{v}) + (\mathbf{u} \cdot \mathbf{w})$$

Vectores: producto cruz

- El producto cruz de dos vectores es un vector que es *perpendicular* a ambos
- El producto cruz está definido como:

$$(\mathbf{v} \times \mathbf{w})_x = v_y w_z - v_z w_y$$
$$(\mathbf{v} \times \mathbf{w})_y = v_z w_x - v_x w_z$$
$$(\mathbf{v} \times \mathbf{w})_z = v_x w_y - v_y w_x$$

Puntos

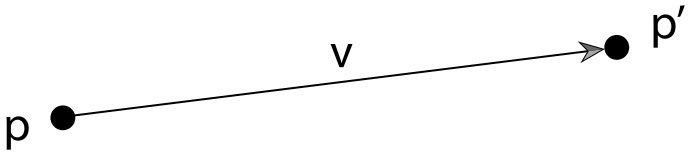
- Es una localidad cero-dimensional en un espacio bi o tridimensional.

- Utilizamos sus tres coordenadas *x, y, z* para denotarlo

$$\mathrm{p} = (p_x, p_y, p_z)^T$$

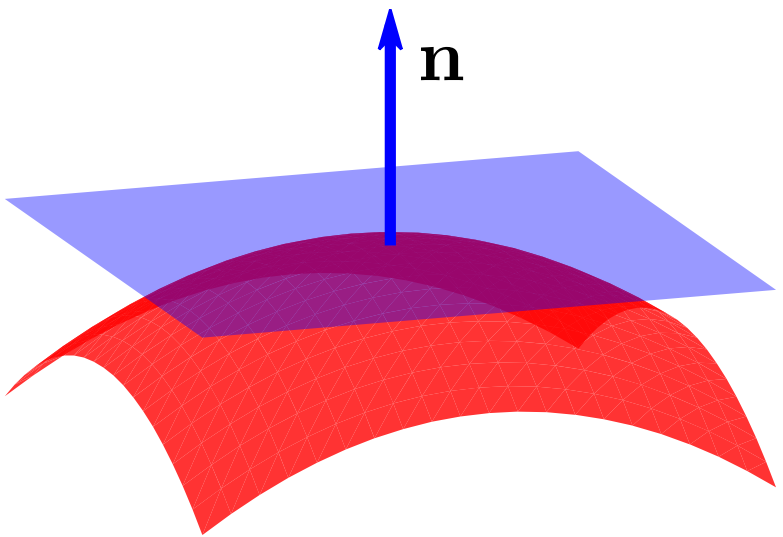
- ¡No confundir con un vector!
  - Aunque sí podemos crear un vector de un punto a otro, utilizando la resta componente a componente de p' con p:

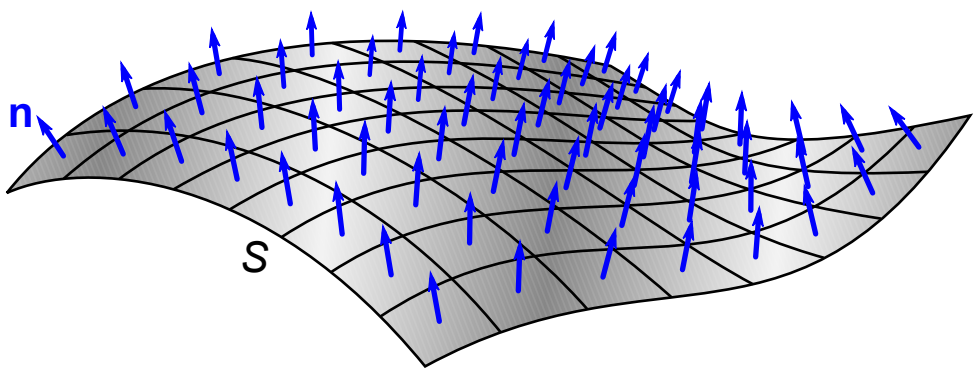
$$\mathbf{v} = \mathrm{p}' - \mathrm{p}$$



Normales

- La normal de una superficie, o sólo normal, es un vector perpendicular a la superficie en una posición particular.
  - ¡No confundir vector normal con vector normalizado!
  - No necesariamente están normalizados

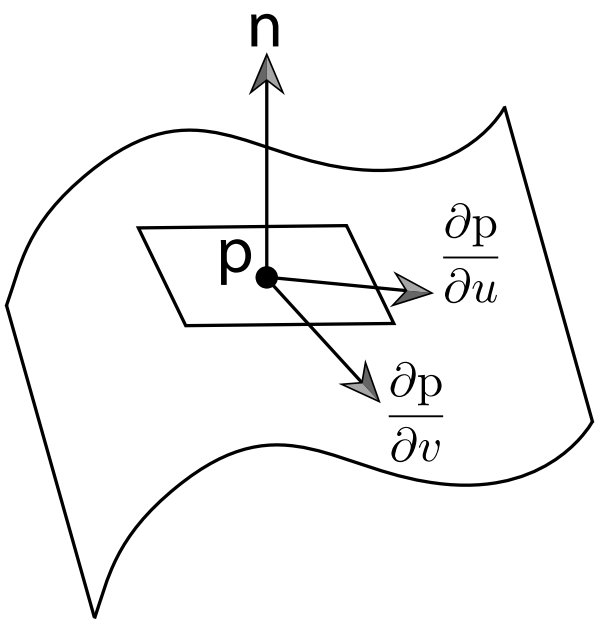




Normales

- Plano perpendicular al punto
  - Derivadas parciales
  - Vectores no nec. ortogonales

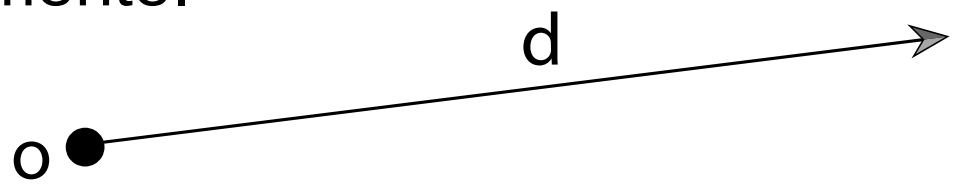
$$\mathbf{n} = \frac{\partial \mathrm{p}}{\partial u} \times \frac{\partial \mathrm{p}}{\partial v}$$



- En la esfera encontrar la normal, se simplifica MUCHO
  - TAREA

Rayos

- Es una línea semi-infinita especificada por un *origen* y una *dirección*: un punto y un vector (normalizado), respectivamente.



- La forma paramétrica de un rayo, nos da el conjunto de *puntos* por los que pasa el *rayo* como una función de t
  - Es un escalar que define la distancia desde el origen del rayo

$$\mathrm{r}(t) = \mathrm{o} + t\mathbf{d}, \quad 0 \leq t < +\infty$$

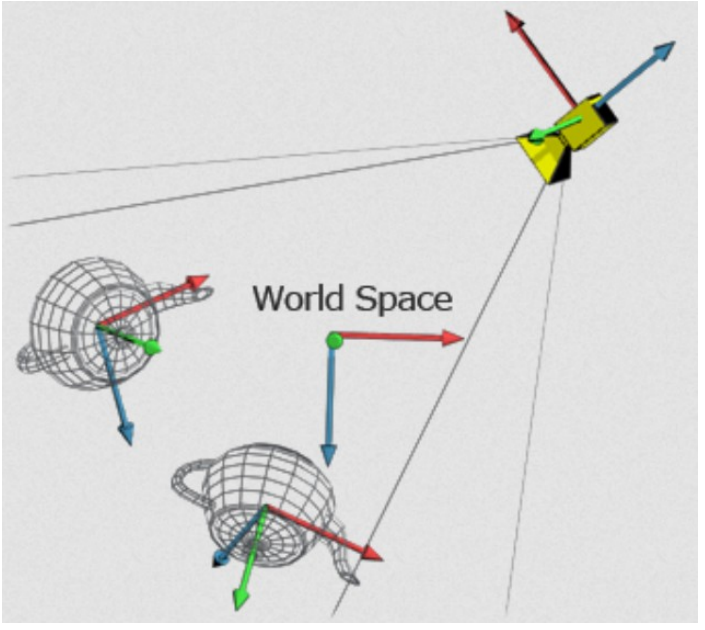
Coordenadas globales

- Son aquellas que se especifican respecto al marco de referencia *global*
  - Es decir, con marco de referencia dado por los ejes x, y, z

- En ocasiones su uso no es ideal:
  - Muchos métodos están definidos respecto a un marco de referencia *local*
    - *Ej. modelos de reflexión y refracción del material*

Coordenadas locales

- Definir un punto o un vector respecto a otro marco de referencia
  - Ej. respecto al punto de intersección en una superficie o respecto a la cámara
- Tenemos ahora tres ejes de referencia $\mathbf{s}, \mathbf{t}, \mathbf{n}$
- Corresponden a los ejes x, y, z *locales*
- Las coordenadas se dan de forma *local*



Coordenadas locales

- Construcción de un sistema de coordenadas locales desde un vector:
  - El primer vector está dado y representará el vector “normal”
  - El segundo es un vector perpendicular al primero, haciendo cero uno de los componentes, invirtiendo los dos restantes y negando uno.
  - El tercer vector es el producto cruz de los dos primeros

Coordenadas locales

- A partir de **n** se construye el sistema
  - Se comparan los componentes para reducir errores en redondeo
  - invLen normaliza

```
void coordinateSystem(const Vector &n, Vector &s, Vector &t) {
    if (std::abs(n.x) > std::abs(n.y)) {
        float invLen = 1.0f / std::sqrt(n.x * n.x + n.z * n.z);
        t = Vector(n.z * invLen, 0.0f, -n.x * invLen);
    } else {
        float invLen = 1.0f / std::sqrt(n.y * n.y + n.z * n.z);
        t = Vector(0.0f, n.z * invLen, -n.y * invLen);
    }
    s = cross(t, n);
}
```

De coordenadas globales a locales

- Un marco de referencia local, está definido por los vectores $\mathbf{s}, \mathbf{t}, \mathbf{n}$ (en coordenadas globales), desde los cuales construimos su matriz de transformación y su inversa:

$$M = \begin{pmatrix} s_x & t_x & n_x & 0 \\ s_y & t_y & n_y & 0 \\ s_z & t_z & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad M^{-1} = M^T = \begin{pmatrix} s_x & s_y & s_z & 0 \\ t_x & t_y & t_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Es una matriz ortogonal (los tres vectores son ortogonales por definición), por lo tanto su inversa es la transpuesta

De coordenadas globales a locales (2)

- Para convertir un punto o vector de coordenadas locales a globales:

$$M\mathbf{v}_{local} = \mathbf{v}_{global}$$

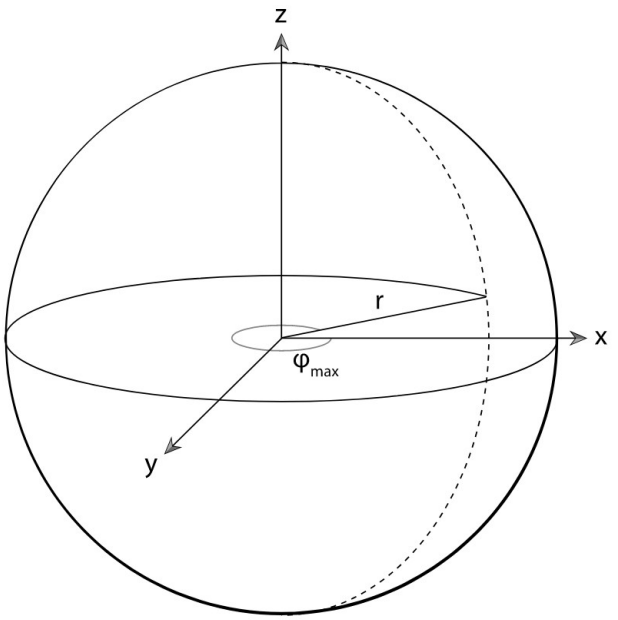
- Análogamente, dado un punto o vector global, obtenemos el equivalente en el espacio local:

$$M^{-1}\mathbf{v}_{global} = \mathbf{v}_{local}$$

- Ojo: se usan coordenadas homogéneas
  - Cuarto componente (o peso) a 1

Esferas

- Es el conjunto de todos los puntos en tres dimensiones que se encuentran a la misma distancia de otro punto.
- ¡Es una superficie!
  - No confundir con “bola”



Esfera (centro y radio)

Esfera unitaria en el origen

- Definida como el conjunto de puntos a una unidad de distancia desde el origen, con la ecuación implícita:

$$x^2 + y^2 + z^2 - 1 = 0$$

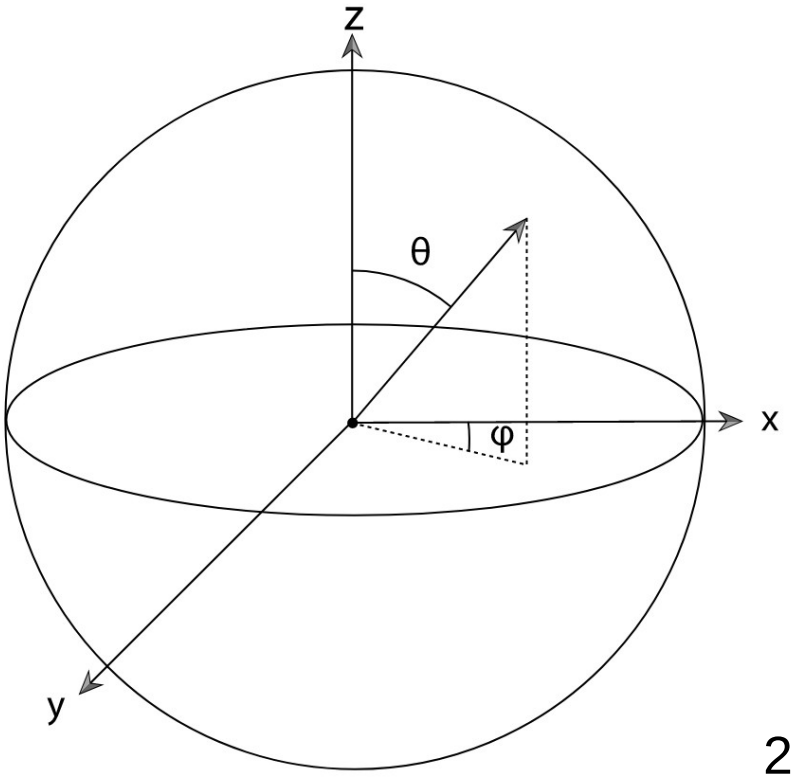
- Sea $\mathbf{v} = \mathrm{p} - \mathbf{0}$, si el vector resultante es unitario (el punto está en la esfera), entonces se satisface:

$$\begin{aligned} \|\mathbf{v}\| &= \sqrt{v_x^2 + v_y^2 + v_z^2} = 1 \\ &= v_x^2 + v_y^2 + v_z^2 = 1 \end{aligned}$$

- Y de este modo, queda demostrada la ecuación implícita

Coordenadas esféricas

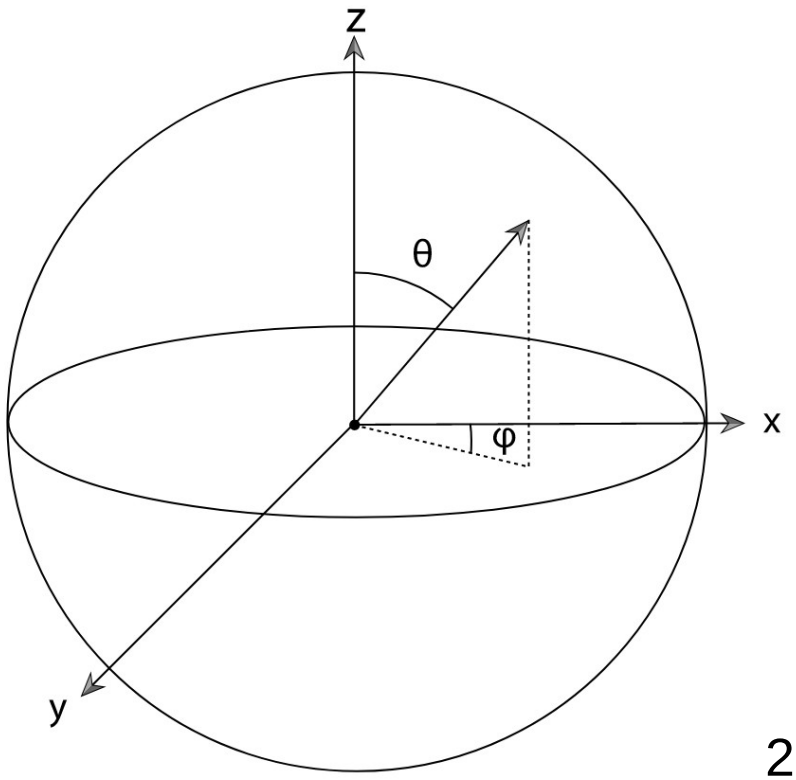
- Análogas a las coordenadas polares
- Se expresan utilizando (r, θ, ϕ)
  - La distancia al origen r
  - El ángulo polar θ respecto a z
    - Definido en $[0, \pi]$
  - El ángulo azimutal ϕ respecto a x
    - Definido en $[0, 2\pi)$



Conversión de coordenadas

- Dadas las coordenadas esféricas, se obtienen las coordenadas cartesianas:

$$x = r \sin\theta \cos\phi$$
$$y = r \sin\theta \sin\phi$$
$$z = r \cos\theta$$



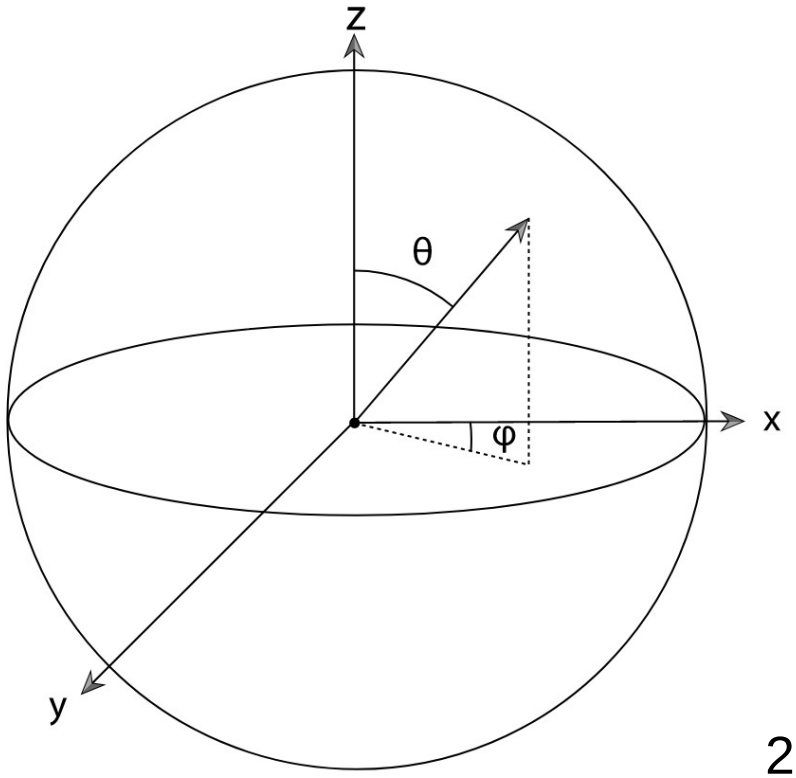
Conversión de coordenadas

- Dadas las coordenadas cartesianas, se obtienen las coordenadas esféricas:

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\theta = \arccos \frac{z}{r}$$

$$\phi = \arctan \frac{y}{x}$$



Intersección rayo-esfera

- Ecuación de una esfera en $\mathbf{p} = (p_x, p_y, p_z)^T$ con radio r:

$$(x - p_x)^2 + (y - p_y)^2 + (z - p_z)^2 - r^2 = 0$$

- Punto sobre un rayo con origen $\mathbf{o}$ y dirección $\mathbf{d}$:

$$\mathbf{o} + t\mathbf{d} = (o_x + td_x, o_y + td_y, o_z + td_z)^T$$

- Para encontrar los puntos sobre el rayo que intersectan la esfera, sustituimos en la ecuación de la esfera:

$$(o_x + td_x - p_x)^2 + (o_y + td_y - p_y)^2 + (o_z + td_z - p_z)^2 - r^2 = 0$$

Intersección rayo-esfera (2)

- Desarrollamos la ecuación y después agrupamos términos para obtener:

$$[d_x^2 + d_y^2 + d_z^2]t^2 + 2[(o_x - p_x)d_x + (o_y - p_y)d_y + (o_z - p_z)d_z]t + (o_x - p_x)^2 + (o_y - p_y)^2 + (o_z - p_z)^2 - r^2 = 0$$

- Esta ecuación cuadrática la reescribimos usando notación de vectores como:

$$(\mathbf{d} \cdot \mathbf{d})t^2 + 2[(\mathbf{o} - \mathbf{p}) \cdot \mathbf{d}]t + (\mathbf{o} - \mathbf{p}) \cdot (\mathbf{o} - \mathbf{p}) - r^2 = 0$$

Intersección rayo-esfera (3)

- Resolviendo la ecuación para t

$$(\mathbf{d}\cdot\mathbf{d})t^2 + 2[(\mathbf{o}-\mathbf{p})\cdot\mathbf{d}]t + (\mathbf{o}-\mathbf{p})\cdot(\mathbf{o}-\mathbf{p}) - r^2 = 0$$

- tenemos:

$$t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

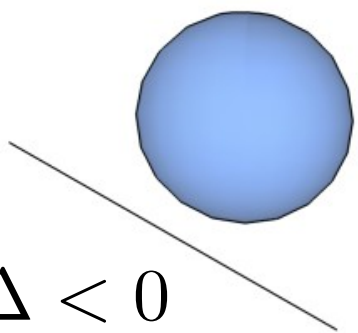
$$a = \mathbf{d}\cdot\mathbf{d}$$
$$b = 2[(\mathbf{o}-\mathbf{p})\cdot\mathbf{d}]$$
$$c = (\mathbf{o}-\mathbf{p})\cdot(\mathbf{o}-\mathbf{p}) - r^2$$

Intersección rayo-esfera (4)

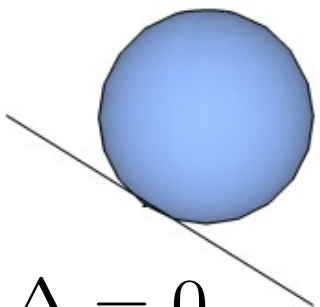
- Ahora, si $\mathbf{d}$ es un vector normalizado, entonces $a = \mathbf{d} \cdot \mathbf{d} = 1$ y la solución es:

$$t = -[(\mathbf{o} - \mathbf{p}) \cdot \mathbf{d}] \pm \sqrt{[(\mathbf{o} - \mathbf{p}) \cdot \mathbf{d}]^2 - (\mathbf{o} - \mathbf{p}) \cdot (\mathbf{o} - \mathbf{p}) + r^2}$$

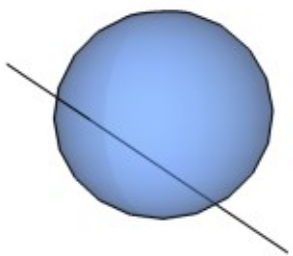
- El discriminante Δ en $\sqrt{\Delta}$ nos indica tres escenarios:



$\Delta < 0$



$\Delta = 0$



$\Delta > 0$

Intersección rayo-esfera (5)

- ¿Qué pasa si el valor del discriminante es positivo pero muy grande?
  - La distancia en el rayo es negativa, ¿cuál es la interpretación geométrica?
  - ¿Estos valores son útiles para nuestro renderizador?

Aceleración de primitivas de intersección

- Big Hero 6

Aceleración de primitivas de intersección

- Coco

Aceleración de primitivas de intersección

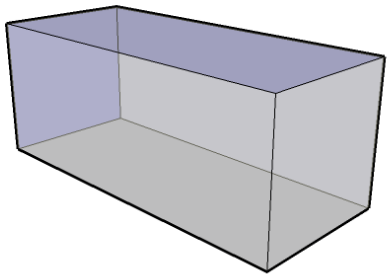
- Una escena puede contener una gran cantidad de objetos
  - Cada objeto con complejidad geométrica arbitraria

- ¿Cómo determinar la primera intersección en un rayo?
  - ¿Calcular la intersección del rayo con *todos* los objetos?
  - ¿Mecanismo *eficiente?*

Bounding boxes

- *Bounding box* literalmente *caja de límites*

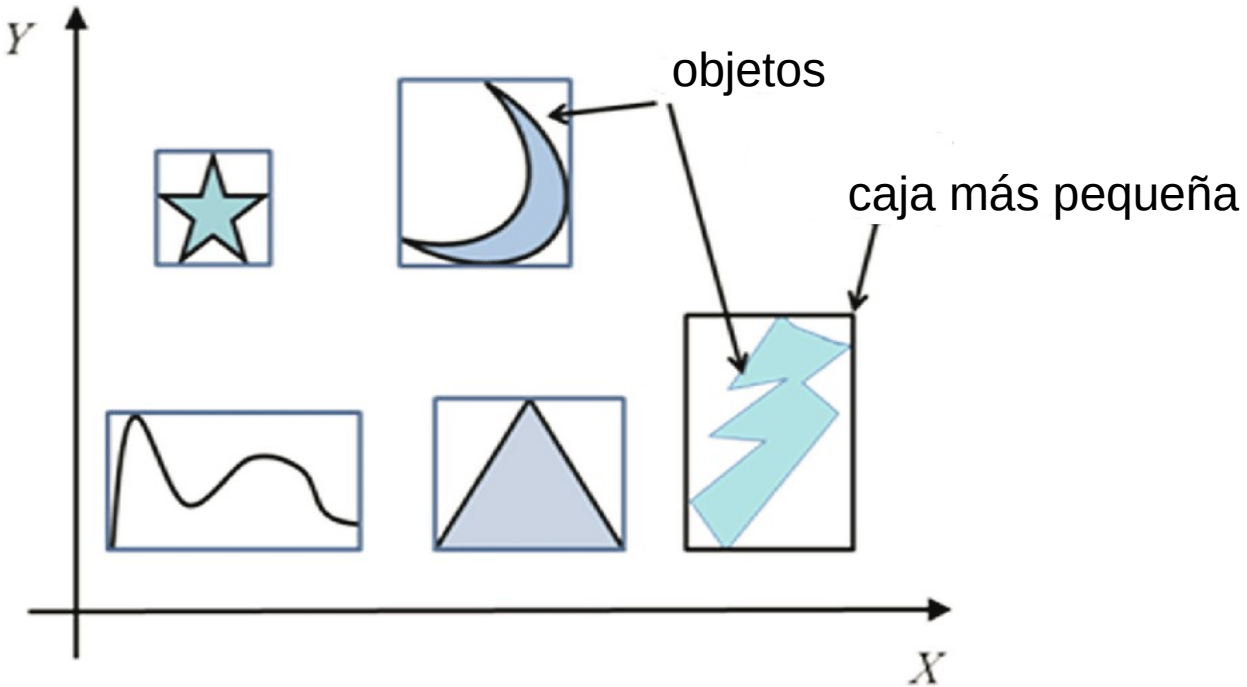
- Es un *ortoedro* (prisma rectangular ortogonal)
  - Cada cara del prisma es un rectángulo



- Nuestra caja *contendrá* el objeto
  - Primitiva de intersección rayo-caja
  - Si hay intersección, entonces rayo-objeto

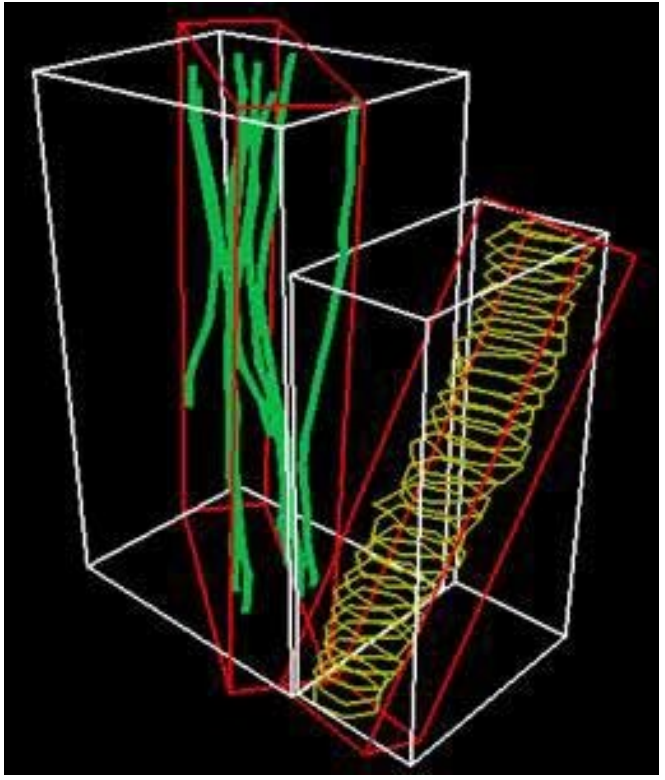
Minimum bounding box

- Es la caja de tamaño mínimo que *contiene* al objeto u objetos



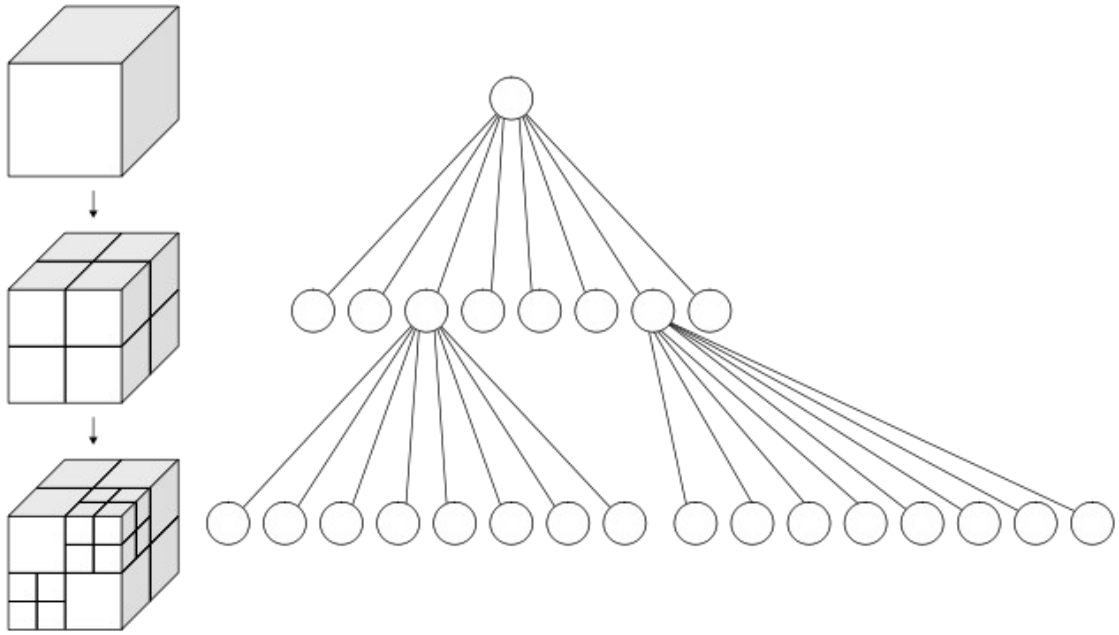
Axis-aligned Bounding Box

- Los objetos pueden formas y transformaciones arbitrarias
- La caja mínima puede no estar alineada con los ejes
- Es deseable poder construir una jerarquía de cajas
  - subdividir *eficientemente* el espacio tridimensional.



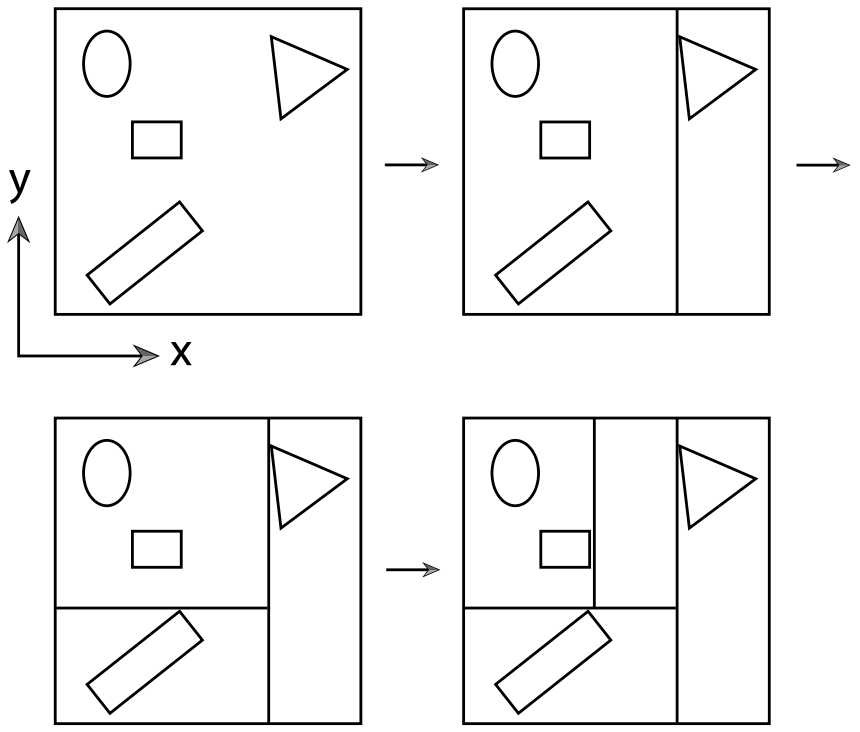
Octree

- La raíz representa el espacio completo
- Cada uno de los nodos es una subdivisión de este espacio
- ¡Cajas alineadas!



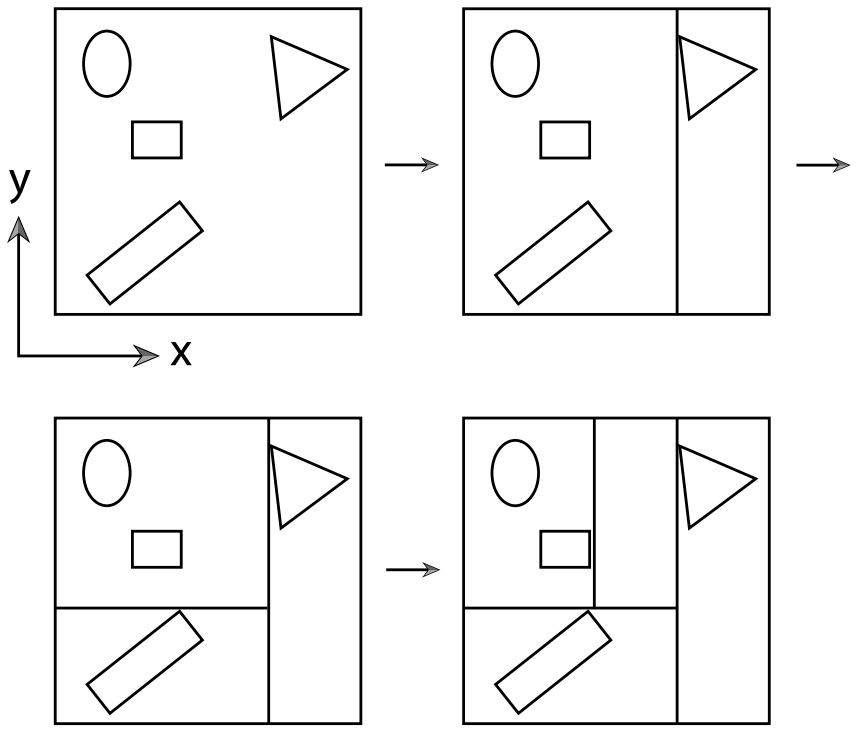
kd-tree

- Es un árbol *binario* que *particiona* un espacio *k* dimensional
  - Cada nodo interior parte en dos la caja utilizando un *plano alineado* a uno de los *ejes*
  - ¡Cajas alineadas!



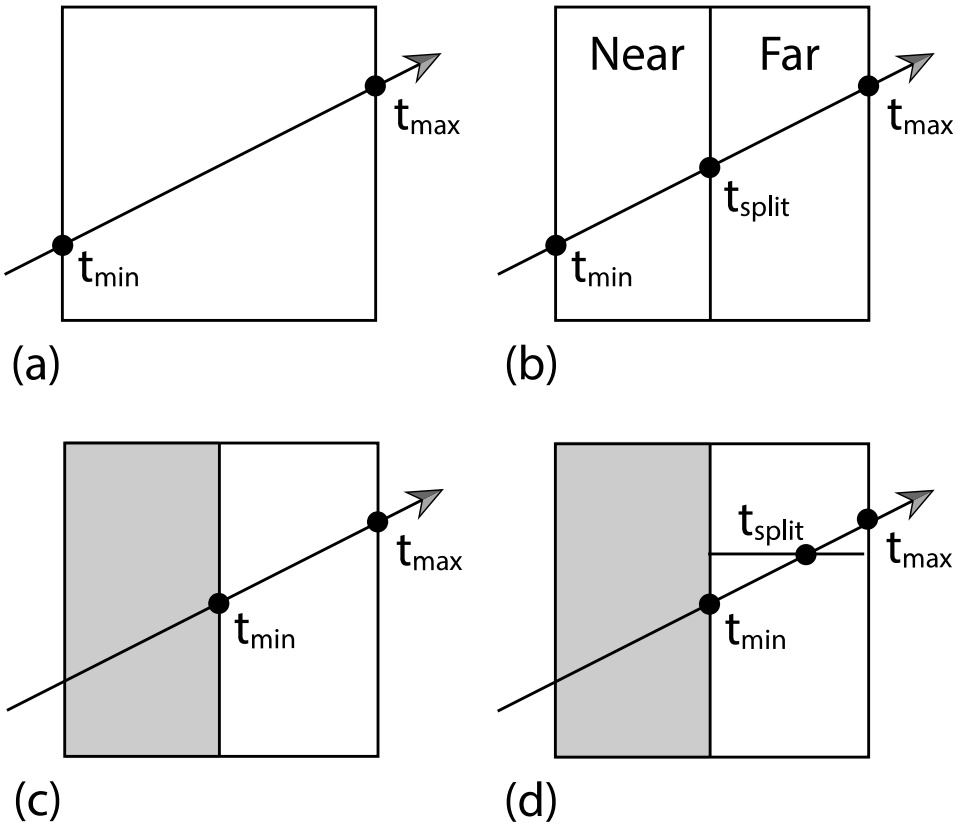
kd-tree (2)

- Cada nodo consiste de:
  - El eje que se parte
  - La posición del corte
  - 2 nodos hijos
- Si es una hoja:
  - Qué primitivas contiene o traslapa
- No utilizar punteros
  - Arreglos
  - Nodos de 8 bytes (cache)



kd-tree (3): atravesar el árbol

- Dado un rayo:
  - (a) Se calculan los dos puntos de intersección con la AABB de la escena (raíz)
  - (b) Se desciende al área cercana
    - Si hay primitivas se calculan
    - Sino procesa los hijos
  - (c) Si no se encuentra una intersección, entonces se procesa el nodo lejano
  - (d) El proceso continúa revisando primero el área cercana, hasta que se encuentre una intersección o el rayo escape.



Intersecciones: perspectiva

- Considere este render:
  - Escena **SIMPLE**
  - Resolución 720x720
  - 4096 spp
  - 14 minutos:
    - Intel Core i7-8700 @ 3.2GHz
    - 6 núcleos, 12 hilos

- **20 mil millones de rayos**