Universidad de Huelva

Tema 10

Orientación espacial

Índice

- Se define 'orientacion' como la configuración de la rotación instantánea de un objeto.
- Piense en ello como la equivalencia rotacional de la posición.

- Las coordenadas Cartesianas (x,y,z) son una forma fácil y natural de representar una posición en el espacio 3D.
- Hay muchas otras alternativas tales como la notación polar (r,θ,φ) y se podría usar otras si queremos.

- ¿Hay una manera sencilla de representar una orientación 3D? (análoga a coordenadas cartesianas?)
- No hay en realidad.
- Hay varias opciones, pensemos en:
  - Angulos de Euler
  - Vectores Rotacion (ejes/angulos)
  - Matrices 3x3
  - Quaternions
  - y mucho mas…

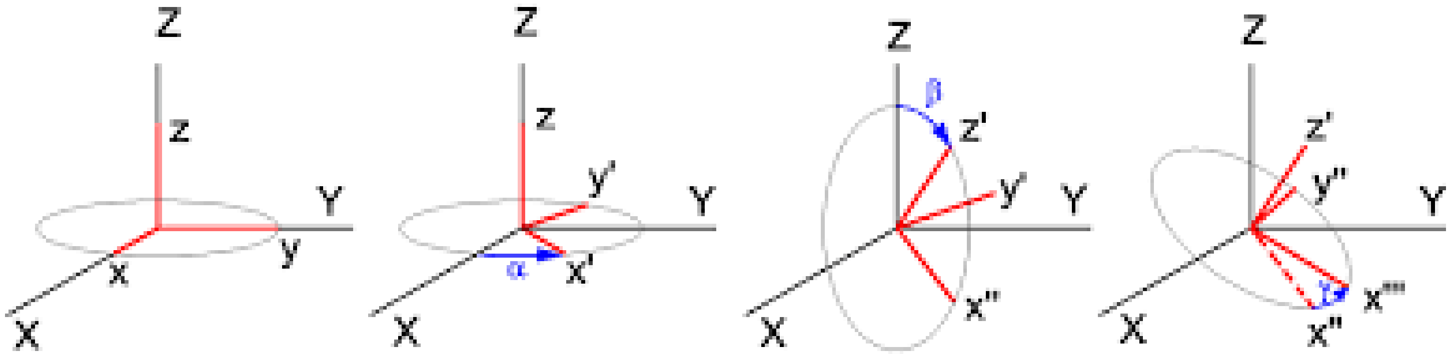
Índice

- Teorema de Euler: Dos sistemas de coordenadas ortonormales independientes pueden relacionarse por una secuencia de rotaciones (no más de tres) alrededor de ejes de coordenadas, donde no hay dos rotaciones sucesivas alrededor del mismo eje.
- No confundir con ángulos de Euler, Integración de Euler, Dinámica de Newton-Euler, Ecuaciones de Euler , característico de Euler…
- Leonard Euler (1707-1783)

- Esto significa que podemos representar una orientación con 3 numeros.
- Una secuencia de rotaciones alrededor de los ejes principales se denomina *Secuencia de Ángulos Euler.*
- Suponiendo que nos limitamos a 3 rotaciones sin rotaciones sucesivas sobre el mismo eje, podríamos utilizar cualquiera de los siguientes 12 secuencias:

XYZ	XZY	XYX	XZX
YXZ	YZX	YXY	YZY
ZXY	ZYX	ZXZ	ZYZ

- Esto nos da 12 maneras redundantes para almacenar una orientación mediante ángulos de Euler.
- No existe un convenio para el manejo de ángulos de Euler. Cada autor o programador utiliza cualquiera de ellos.



- Para construir una matriz a partir de un conjunto de ángulos de Euler, vamos a multiplicarlo por una secuencia de matrices de rotación juntas:

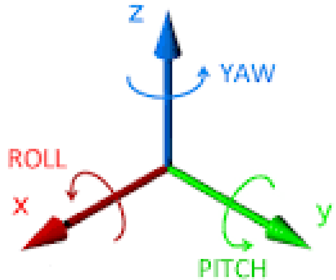
$$\mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_x & s_x \\ 0 & -s_x & c_x \end{bmatrix} \cdot \begin{bmatrix} c_y & 0 & -s_y \\ 0 & 1 & 0 \\ s_y & 0 & c_y \end{bmatrix} \cdot \begin{bmatrix} c_z & s_z & 0 \\ -s_z & c_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

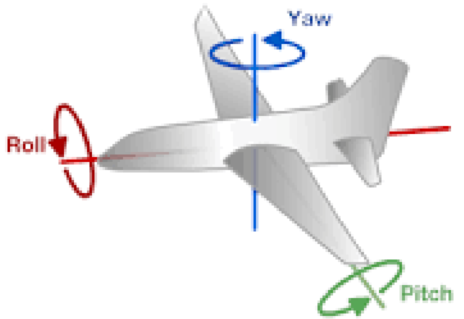
$$= \begin{bmatrix} c_y c_z & c_y s_z & -s_y \\ s_x s_y c_z - c_x s_z & s_x s_y s_z + c_x c_z & s_x c_y \\ c_x s_y c_z + s_x s_z & c_x s_y s_z - s_x c_z & c_x c_y \end{bmatrix}$$

- Como la multiplicación de matrices no es conmutativa, el orden de las operaciones es importante.
- Se asume que las rotaciones son relativa a los ejes fijos mundo, en lugar de local para el objeto.

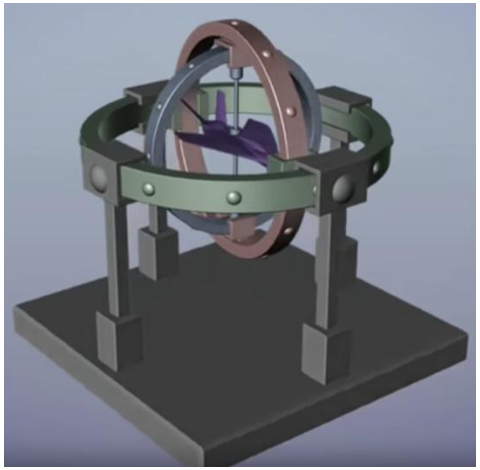
- Para usar ángulos de Euler, hay que elegir cuál de las 12 representaciones usar.
- Puede haber algunas diferencias prácticas entre ellos y la mejor secuencia puede depender de qué es exactamente lo que se está tratando de lograr.

- En general, para vehículos (aviones, coches), suele utilizarse la rotación *roll-pitch-yaw*.
- En situaciones donde hay un plano de tierra determinado, los Ángulos de Euler pueden realmente ser una representación intuitiva.





- Un problema potencial que se puede sufrir es el bloqueo de ejes o ***gimbal lock***.
- Esto se produce cuando dos ejes efectivamente se alinean, lo que resulta en una pérdida temporal de un grado de libertad.



https://www.youtube.com/watch?v=zc8b2Jo7mno

- Para animar un objeto se puede simplemente interpolar entre los tres valores de forma independiente.
- Esto dará como resultado que la interpolación siga un camino diferente dependiendo de cuál de los 12 esquemas se elige
- Esto puede ser o no un problema, dependiendo de la situación.
- Interpolar cerca de los "polos" puede ser problemático.
- Nota: cuando se interpole ángulos, recuerde revisar el cruce en las fronteras de +180 / -180 grados.

- Los Ángulos de Euler se utilizan en una gran cantidad de aplicaciones, pero tienden a requerir algunas decisiones no arbitrarias.
- Tampoco interpolan de una forma consistente (pero esto no siempre es malo).
- Pueden sufrir del bloqueo de ejes y otros problemas relacionados.
- No hay un camino simple para concatenar rotaciones.
- La conversion hacia/desde una matriz requiere varias operaciones trigonometricas.
- Son compactos (requieren solo 3 numeros).

- El Teorema de Euler tambien demuestra que cualquiera dos rotaciones pueden ser relacionadas por una rotación simple sobre algunos ejes (no necesariamente un eje principal).
- Esto significa que se puede representar una orientación arbitraria como una rotación sobre ejes unidad por algunos angulos (4 numeros) (Forma Ejes/Angulo)
- Alternativamente, podemos escalar los ejes por el angulo y compactarlo en un solo vector 3D (vector Rotación).

- La matriz asociada a una rotación θ alrededor de un eje unidad arbitrario **a** es la siguiente:

$$\begin{bmatrix} a_x^2 + c_\theta(1-a_x^2) & a_x a_y(1-c_\theta) + a_z s_\theta & a_x a_z(1-c_\theta) - a_y s_\theta \\ a_x a_y(1-c_\theta) - a_z s_\theta & a_y^2 + c_\theta(1-a_y^2) & a_y a_z(1-c_\theta) + a_x s_\theta \\ a_x a_z(1-c_\theta) + a_y s_\theta & a_y a_z(1-c_\theta) - a_x s_\theta & a_z^2 + c_\theta(1-a_z^2) \end{bmatrix}$$

- Para convertir un vector de rotación escalado a matriz, se tendría que extraer la magnitud de esta y rotar alredededor de los ejes normalizados.
- Normalmente, el formato de vector de rotación es más útil para la representación de velocidades y aceleraciones angulares, en lugar de posición angular (orientación).

- El almacenamiento de una orientación como un eje y un ángulo utiliza 4 números, pero el teorema de Euler dice que sólo tenemos 3 números para representar una orientación.
- Matemáticamente, esto significa que estamos utilizando 4 grados de libertad para representar un valor de 3 grados de libertad. Esto implica que hay información redundante en el formato de eje/ángulo.
- La redundancia se manifiesta en el módulo del vector de eje. El módulo no lleva ninguna información, por lo que es redundante. Para eliminar la redundancia, elegimos normalizar el eje, lo cual limita el grado de libertad adicional.

- Se puede usar una matriz 3x3 para representar una orientación.
- Eso significa que ahora tenemos 9 números en vez de 3, y por tanto, tenemos 6 grados de libertad extra.
- NOTA: No se usa matrices 4x4 aquí, como sería seguramente mas útil debido a que nos da la habilidad de combinar traslaciones. No hablaremos de traslaciones por lo que pensemos en matrices 3x3.

- Esos 6 GDL extra se manifiestan como 3 escalas (x,y e z) y 3 cortes (xy, xz, and yz)
- Si asumimos que la matriz representa una transformación rígida (ortonormal), entonces podemos restringir los 6 GDL extras.

$$\begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{bmatrix}$$

$$|\mathbf{a}| = |\mathbf{b}| = |\mathbf{c}| = 1$$

$$\mathbf{a} = \mathbf{b} \times \mathbf{c}$$

$$\mathbf{b} = \mathbf{c} \times \mathbf{a}$$

$$\mathbf{c} = \mathbf{a} \times \mathbf{b}$$

- Las matrices son por lo general la forma computacionalmente más eficiente para aplicar rotaciones a los datos geométricos, y por lo tanto la mayoría de las representaciones de orientación necesitan en última instancia, ser convertidos en una matriz con el fin de hacerla útil (transformar vértices ...)
- ¿Por qué entonces, no deberíamos simplemente siempre usar matrices?
  - Cuestines numéricas.
  - Cuestiones de almacenamiento.
  - Problemas de ínteraccion de los usuarios.
  - Problemas de interpolación.

- Los quaterniones son un concepto matemático interesante con una profunda relación con los fundamententos de algebra lineal y la teoría de numeros.
- Inventados por W.R.Hamilton en 1843.
- En la práctica, son muy útiles para nosotros como medio para representar orientaciones.
- Un quaternion tiene 4 componentes.

$$\mathbf{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}$$

- Los Quaterniones son en realidad una extensión de los numeros complejos.
- De los 4 componentes, uno es un numero escalar ‘real’ y los otros 3 forman un vector en el espacio ijk imaginario!

$$\mathbf{q} = q_0 + iq_1 + jq_2 + kq_3$$

$$i^2 = j^2 = k^2 = ijk = -1$$

$$i = jk = -kj$$

$$j = ki = -ik$$

$$k = ij = -ji$$

- Algunas veces, se escriben como combinación de un valor escalar *s* y un valor vector *v*

$$\mathbf{q} = \langle s, \mathbf{v} \rangle$$

donde

$$s = q_0$$
$$\mathbf{v} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}$$

- Por conveniencia, se usarán solamente quaterniones de longitud unidad, ya que serán suficientes para nuestros propositos y nos harán las cosas un poco más fáciles

$$|\mathbf{q}| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} = 1$$

- Estos corresponden a la serie de vectores que forman la "superficie" de una hiperesfera 4D de radio unidad
- La "superficie" es en realidad un volumen 3D en el espacio 4D, pero a veces puede ser visualizado como una extensión para el concepto de una superficie 2D sobre una esfera 3D.

- Un quaternion puede representar una rotación por un angulo θ alrededor de un eje unidad a:

$$\mathbf{q} = \left[\cos\frac{\theta}{2} \quad a_x \sin\frac{\theta}{2} \quad a_y \sin\frac{\theta}{2} \quad a_z \sin\frac{\theta}{2} \right]$$

o

$$\mathbf{q} = \left\langle \cos\frac{\theta}{2}, \mathbf{a} \sin\frac{\theta}{2} \right\rangle$$

- Si a es de longitud unidad, entonces q también

$$
\begin{aligned}
|\mathbf{q}| &= \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \\
&= \sqrt{\cos^2\frac{\theta}{2} + a_x^2 \sin^2\frac{\theta}{2} + a_y^2 \sin^2\frac{\theta}{2} + a_z^2 \sin^2\frac{\theta}{2}} \\
&= \sqrt{\cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}\left(a_x^2 + a_y^2 + a_z^2\right)} \\
&= \sqrt{\cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}|\mathbf{a}|^2} = \sqrt{\cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}} \\
&= \sqrt{1} = 1
\end{aligned}
$$

- Para convertir un quaternion a una matriz rotación:

$$\begin{bmatrix} 1-2q_2^2-2q_3^2 & 2q_1q_2+2q_0q_3 & 2q_1q_3-2q_0q_2 \\ 2q_1q_2-2q_0q_3 & 1-2q_1^2-2q_3^2 & 2q_2q_3+2q_0q_1 \\ 2q_1q_3+2q_0q_2 & 2q_2q_3-2q_0q_1 & 1-2q_1^2-2q_2^2 \end{bmatrix}$$

- Para pasar de una matriz a Quaternion, habría que hacer algunas operaciones.
- Estas incluye un algoritmo con algunos ‘If’, una raiz cuadrada, tres divisiones, y algunas otras cosas
- Ver el libro de Sam Buss (p.305) para el algoritmo.

- Pensemos en una persona que está encima de una esfera gigantesca (como un planeta).
- Desde el punto de vista de la persona, se puede mover en dos ejes ortogonales (delante/atras) e (izquierda/derecha)
- No hay percepción de los polos fijos o longitud / latitud, porque no importa en qué dirección se mueva, siempre tienen dos formas ortogonales para ir
- Desde su punto de vista, podría estar moviéndose en un plano 2D infinito, sin embargo, si van demasiado lejos en una dirección, ¡volverá al punto de partida!

- Ahora extendemos este concepto para movernos en una hiperesfera de quaternions unitarios.
- La persona que ahora tiene tres direcciones ortogonales para ir.
- No importa la forma en que se orientan en este espacio, que siempre pueden moverse en alguna combinación de marcha adelante / atrás, izquierda / derecha y arriba / abajo.
- Si van demasiado lejos en alguna dirección, van a volver al punto de partida.

- Consideremos ahora que la ubicación de una persona en esta hiperesfera representa una orientación.
- Cualquier desplazamiento incremental a lo largo de uno de los ejes ortogonales en el espacio curvado corresponde a una rotación incrementales a lo largo de un eje en el espacio real (distancias a lo largo de la hiperesfera corresponden a ángulos en el espacio 3D)
- Moverse en una dirección arbitraria corresponde a girar en torno a algunos ejes arbitrarios. Si usted se mueve demasiado lejos en una dirección vuelve al punto de partida (que corresponde a la rotación de 360 grados alrededor de uno de los ejes)

- Una distancia de x a lo largo de la superficie de la hiperesfera corresponde a una rotación de ángulo 2x radianes.
- Esto significa que si nos movemos en un arco de 90 grados en la hiperesfera corresponde a girar un objeto por 180 grados.
- Viajar 180 grados corresponde a una rotación de 360 grados, por lo que conseguimos volver de nuevo al punto de partida.
- Esto implica que q y -q corresponden a la misma orientación.

- Considere lo que sucedería si no fuera verdad lo que acabamos de decir, y por tanto 180 grados a lo largo de la hiperesfera corresponden a una rotación de 180 grados.
- Esto quiere decir que hay exactamente una orientación que es 180 opuesta a una orientación de referencia.
- En realidad, hay un continuo de posibles orientaciones que son 180 desde una referencia.
- Se pueden encontrar en el ecuador con respecto a cualquier punto de la hiperesfera.

- Considera también qué pasa si se gira un libro 180 alrededor de X, después 180 alrededor de Y, a continuación, 180 en torno a Z. ¡Se termina donde empezó!
- Esto corresponde a viajar a lo largo de un triángulo en la hiperesfera donde cada borde es un arco de 90 grados, ortogonales entre sí a cada borde.

- El producto escalar de dos quaterniones es equivalente al producto escalar de dos vectores:

$$\mathbf{p} \cdot \mathbf{q} = p_0 q_0 + p_1 q_1 + p_2 q_2 + p_3 q_3 = |\mathbf{p}||\mathbf{q}| \cos \varphi$$

- El angulo entre dos quaternions en el espacio 4D es la mitad del angulo que se necesitaria para rotar desde una orientación a otra en el espacio 3D

- Podemos ejecutar multiplicaciones de quaterniones si se pone este en forma de numero complejo

$$\mathbf{q} = q_0 + iq_1 + jq_2 + kq_3$$

- Si **q** representa una rotación y **q′** representa otra rotación, entonces **qq′** representa **q** rotada por **q′**
- Esto sigue reglas muy similares a los de multiplicación de matrices (es decir, no conmutativo)

$$\mathbf{qq}' = (q_0 + iq_1 + jq_2 + kq_3)(q'_0 + iq'_1 + jq'_2 + kq'_3)$$
$$= \langle ss' - \mathbf{v}\cdot\mathbf{v}', s\mathbf{v}' + s'\mathbf{v} + \mathbf{v}\times\mathbf{v} \rangle$$

- Dos quaterniones unitarios multiplicados da como resultado otro quaternion unitario.
- Esto cumple la misma propiedad que los numeros complejos.
- Recordar que la multiplicación por numeros complejos puede ser vista como una rotación en el plano complejo.
- Los quaterniones extienden las rotaciones planas de los números complejos a las rotaciones en el espacio 3D .

- Se puede crear un esqueleto usando articulaciones quaternion.
- Una posibilidad es simplemente permitir un tipo de articulación quaternion y proporcionar una función de matriz local que tome un quaternion.
- Otra posibilidad es también calcular las matrices mundo como multiplicaciones quaternions. Esto implica un poco menos de matematicas que las matrices, pero puede no ser suficientemente rápido. De todas formas, todavía habría que manejar los desplazamientos de las articulaciones (offset) con matrices.

- Usar quaterniones en el esqueleto añade algunas complicaciones, ya que no pueden simplemente ser tratados como 4 GDLs independientes. La razón es que los 4 números no son independientes, y por lo que un sistema de animación tendría que manejarse de forma específica como un quaternion.
- Para hacer frente a esto, uno podría tener que ampliar el concepto de vector pose conteniendo un conjunto de escalares y un array de quaterniones.
- Cuanto mas alto es el nivel de animación con código de mezclas y manipulación de poses, se tendrá que tratar quaterniones de forma especial.

- Una interpolación lineal entre dos puntos a y b en el espacio normal se calcula como

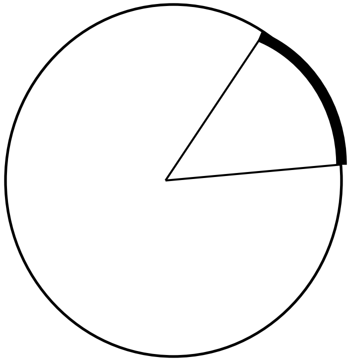
$$\mathrm{Lerp}(t,a,b) = (1-t)\,\mathrm{a} + (t)\,\mathrm{b}$$

donde t va desde 0 a 1

- La operación Lerp puede ser vista como un promedio ponderado.
- También se podría escribir en su forma aditiva de la siguiente forma:

$$\mathrm{Lerp(t,a,b) = a + t(b-a)}$$

- Si queremos interpolar entre dos puntos sobre una esfera (o hiperesfera), no solo queremos realizar Lerp entre los puntos
- En su lugar, vamos a viajar a través de la superficie de la esfera, siguiendo un "gran arco"



- Definiremos la interpolación esférica lineal entre dos vectores unitarios en un espacio N dimensional como:

$$Slerp(t,\mathbf{a},\mathbf{b}) = \frac{\sin\left(\left(1-t\right)\theta\right)}{\sin\theta}\mathbf{a} + \frac{\sin\left(t\theta\right)}{\sin\theta}\mathbf{b}$$

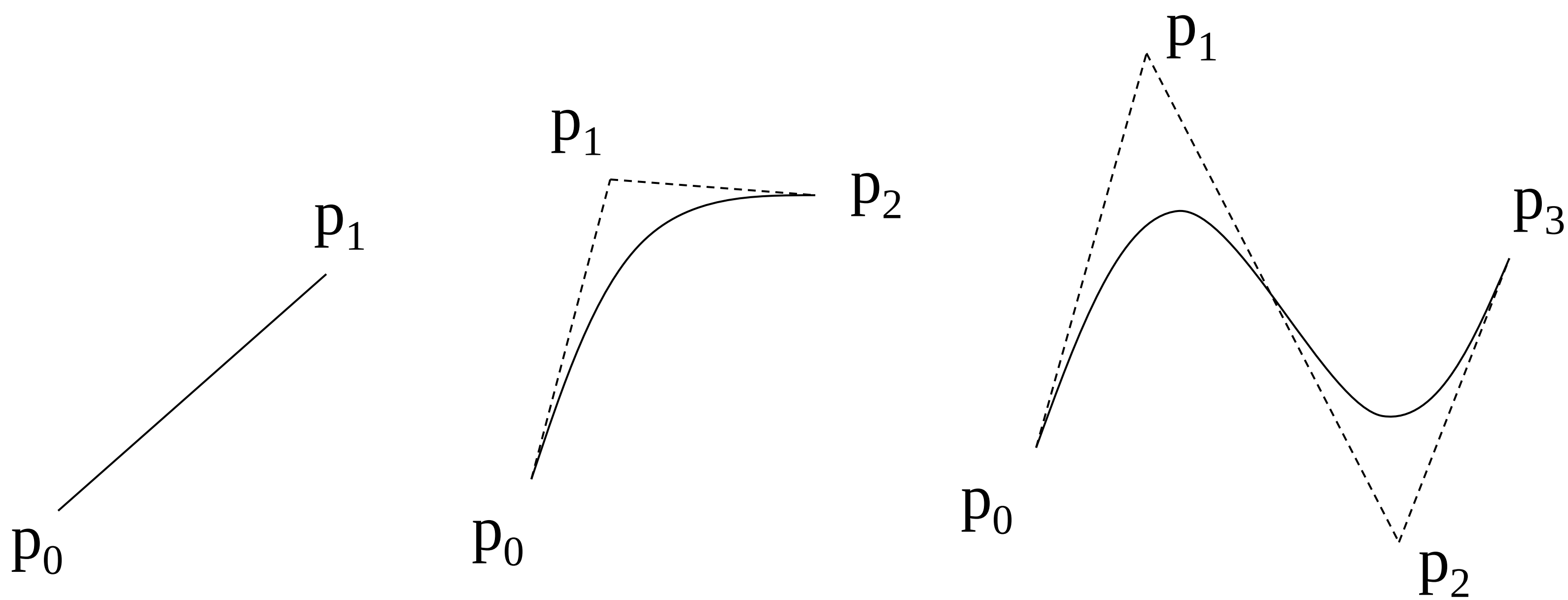
$$donde : \theta = \cos^{-1}\left(\mathbf{a}\cdot\mathbf{b}\right)$$

- Recordar que hay dos vectores redundantes en el espacio de quaternion para cada orientación única en el espacio 3D.
- Cual es la diferencia entre:

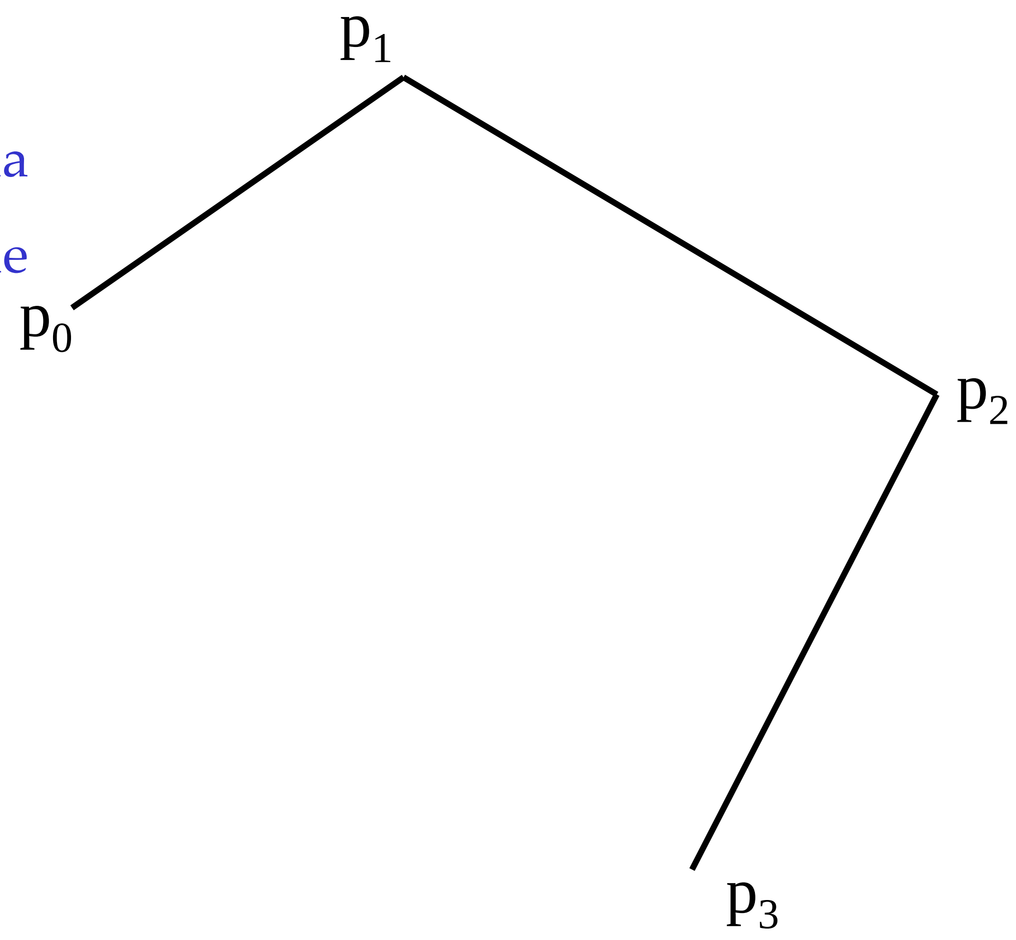
Slerp(t,a,b) y Slerp(t,-a,b) ?

- Uno de ellos será viajar menos de 90 grados, mientras que el otro será viajar más de 90 grados a través de la esfera.
- Esto corresponde a la rotación del "camino corto" o "largo camino”.
- Por lo general, queremos tomar el camino más corto, así que nos negamos a uno de ellos si su producto escalar es <0.

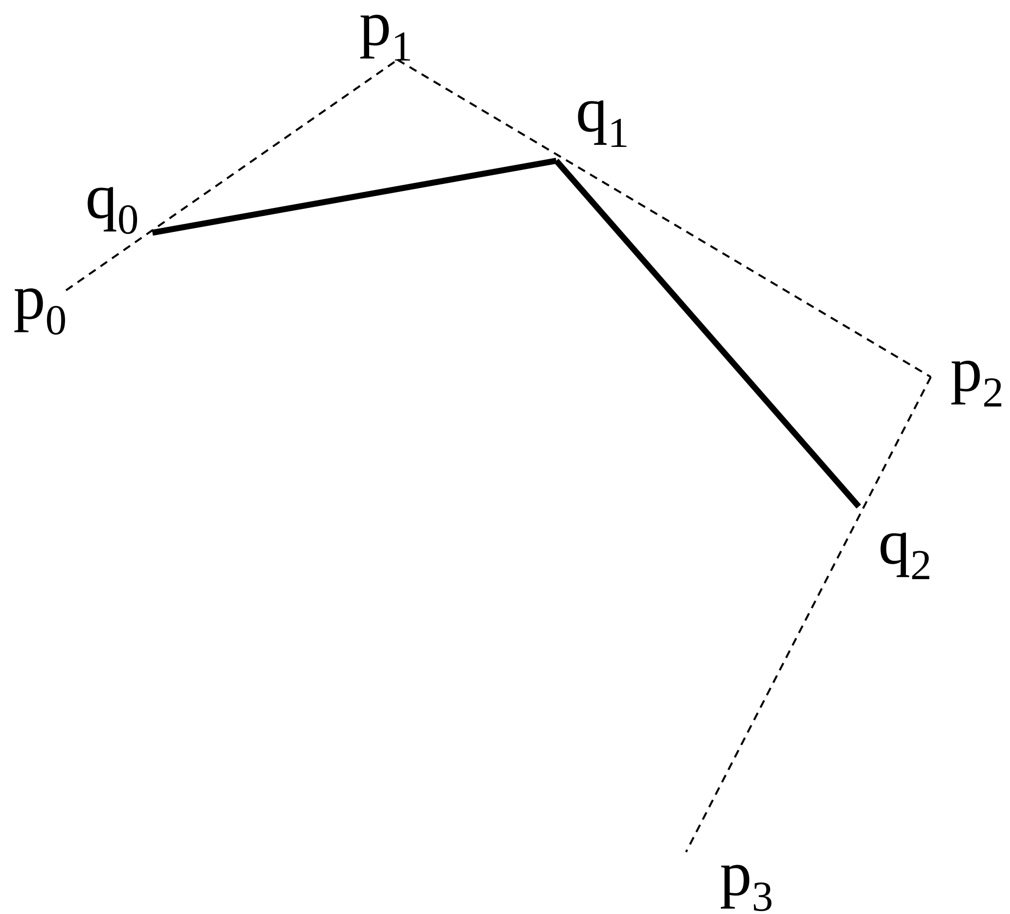
- Las curvas de Bezier pueden ser vistas como una extensión de mayor orden de la interpolación lineal.



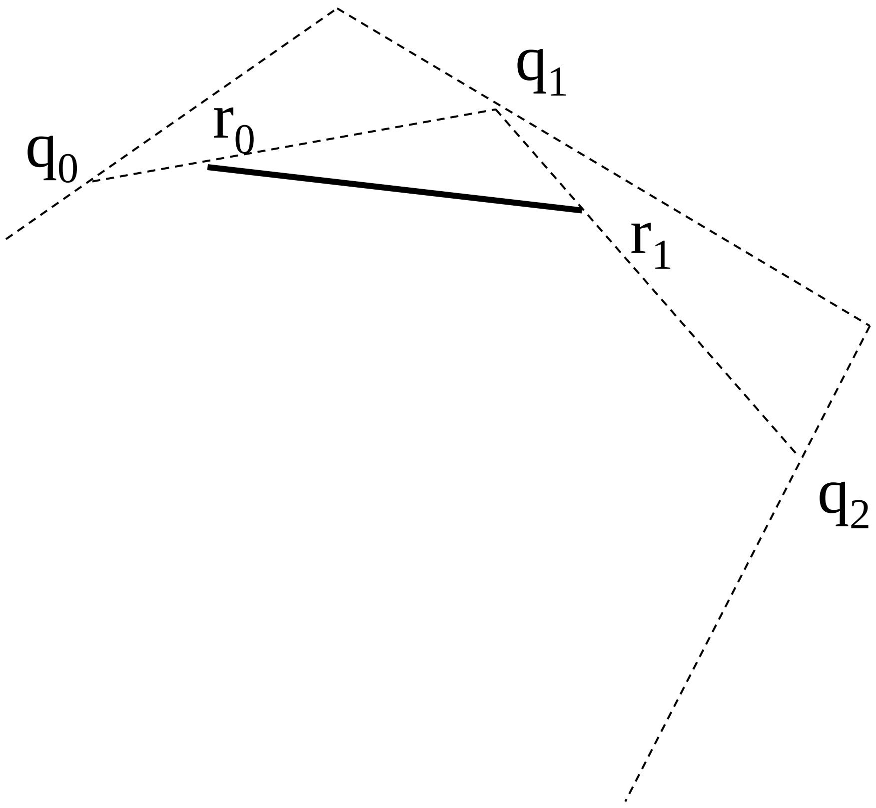
- Encontrar el punto x sobre la curva como una función de parámetro t:



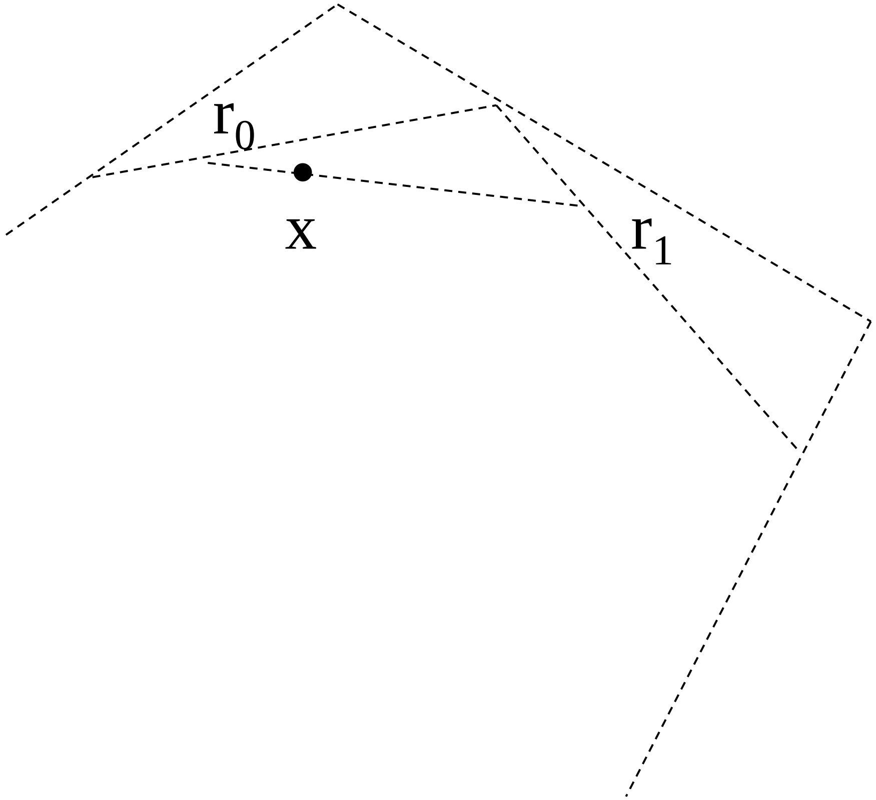
$$\mathbf{q}_0 = Lerp(t, \mathbf{p}_0, \mathbf{p}_1)$$
$$\mathbf{q}_1 = Lerp(t, \mathbf{p}_1, \mathbf{p}_2)$$
$$\mathbf{q}_2 = Lerp(t, \mathbf{p}_2, \mathbf{p}_3)$$

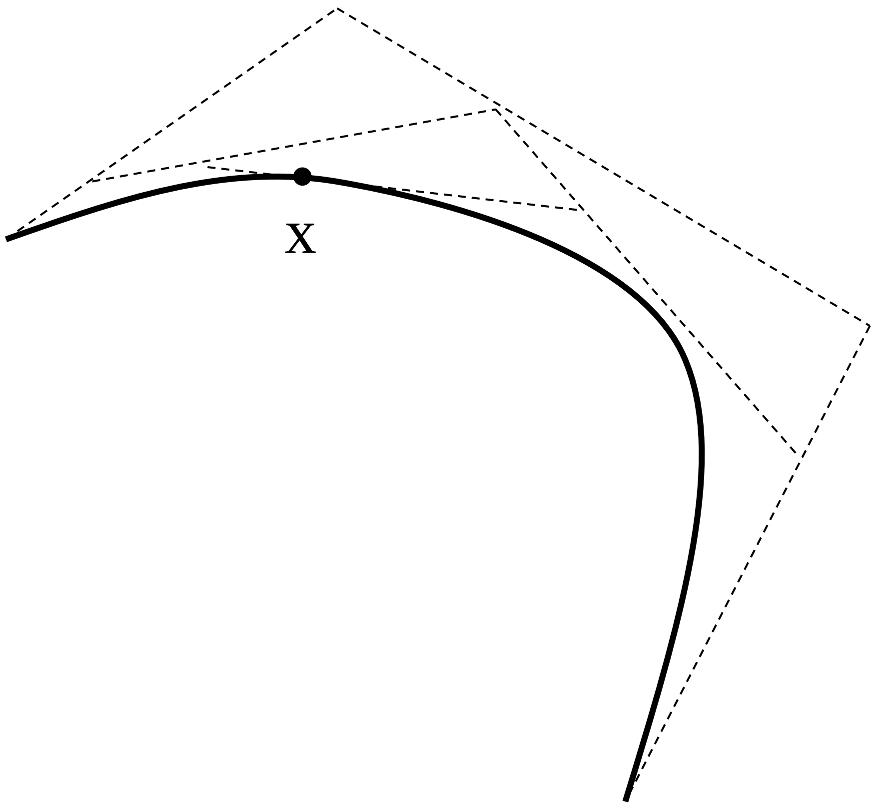


$$\mathbf{r}_0 = Lerp(t, \mathbf{q}_0, \mathbf{q}_1)$$
$$\mathbf{r}_1 = Lerp(t, \mathbf{q}_1, \mathbf{q}_2)$$



$$\mathbf{x} = Lerp(t, \mathbf{r}_0, \mathbf{r}_1)$$




x

- Algoritmo de *de Castlejau*:

$$\mathbf{x} = Lerp(t, \mathbf{r}_0, \mathbf{r}_1) \quad \begin{aligned} \mathbf{r}_0 &= Lerp(t, \mathbf{q}_0, \mathbf{q}_1) \\ \mathbf{r}_1 &= Lerp(t, \mathbf{q}_1, \mathbf{q}_2) \end{aligned} \quad \begin{aligned} \mathbf{q}_0 &= Lerp(t, \mathbf{p}_0, \mathbf{p}_1) \\ \mathbf{q}_1 &= Lerp(t, \mathbf{p}_1, \mathbf{p}_2) \\ \mathbf{q}_2 &= Lerp(t, \mathbf{p}_2, \mathbf{p}_3) \end{aligned} \quad \begin{aligned} &\mathbf{p}_0 \\ &\mathbf{p}_1 \\ &\mathbf{p}_2 \\ &\mathbf{p}_3 \end{aligned}$$

- Podemos contruir curvas de Bezier en la hiperesfera 4D siguiendo exactamente el mismo procedimiento, usando Slerp en vez de Lerp.
- Es una buena idea para invertir (negar) los quaterniones de entrada cuando sea necesario con el fin de hacer que se vaya el "camino corto“.
- Hay otros algoritmos de interpolación más sofisticados, que se pueden aplicar a una hiperesfera
  - Interpolate several key poses
  - Más control sobre la velocidad angular, aceleración angular, suavidad ...

- Quaternions son vectores 4D que pueden representar orientaciones de cuerpos rigidos en 3D
- Forzamos a que sean de longitud unidad
- Claves en la funcion de animation:
  - Quaternion-a-matriz / matriz-a-quaternion
  - Multiplicacion de Quaternion: mas rápido que la multiplicacion de matrices
  - Slerp: interpolar entre orientaciones arbitrarias
  - Curvas Esfericas: de Castlejau algorithm para curvas de Bezier, curvas sobre hiperesferas