Mathématiques pour la 3D

Présenté par :
Bart GEORGE
EISTI
ING3 – Option Visual Computing

Sources pour ce cours

- Fletcher Dunn & Ian Parberry : *3D Math Primer for Graphics and Game Development* (Worldware, 2002, 2ème éd. CRC Press 2011)
- Jason Gregory : *Game Engine Architecture* (AK Peters, 2009, 2ème éd. CRC Press 2014)
- R. Stuart Ferguson : Practical Algorithms for 3D Computer Graphics (2ème ed. CRC Press 2014)
- Cours de Rémi Ronfard, INRIA : https://team.inria.fr/imagine/remi-ronfard/remi-ronfard-teaching/

Deuxième partie

Coordonnées polaires
Rotations et orientation en 3D

Plan

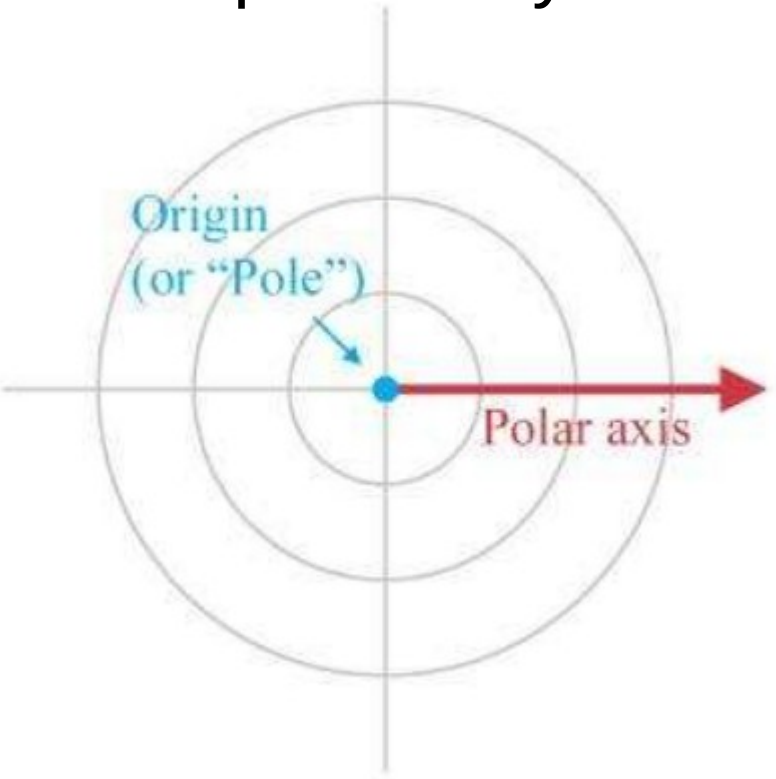
- Coordonnées polaires
- Rotations et orientation en 3D

Plan

- Coordonnées polaires
  - Espace de coordonnées polaires en 2D
  - A quoi servent les coordonnées polaires
  - Espace de coordonnées polaires en 3D
    - Coordonnées 3D cylindriques
    - Coordonnées 3D sphériques
- Rotations et orientation en 3D

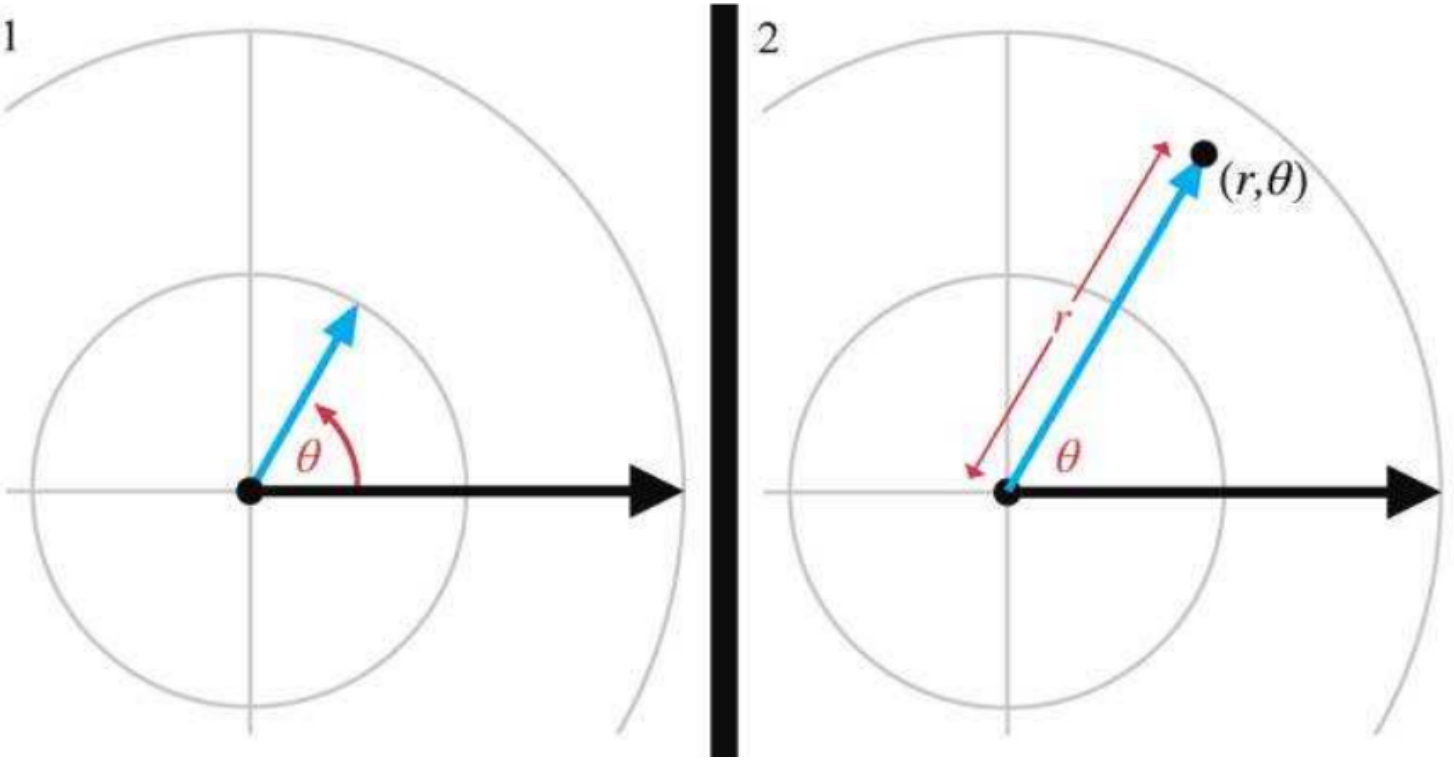
Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Une origine (ou un **pôle**) qui définit le centre de l'espace de coordonnées
  - Un seul axe, dit **polaire**, représenté par un rayon partant de l'origine
    - En général
      - On le fait partir à droite
      - Il représente l'axe des x du système cartésien



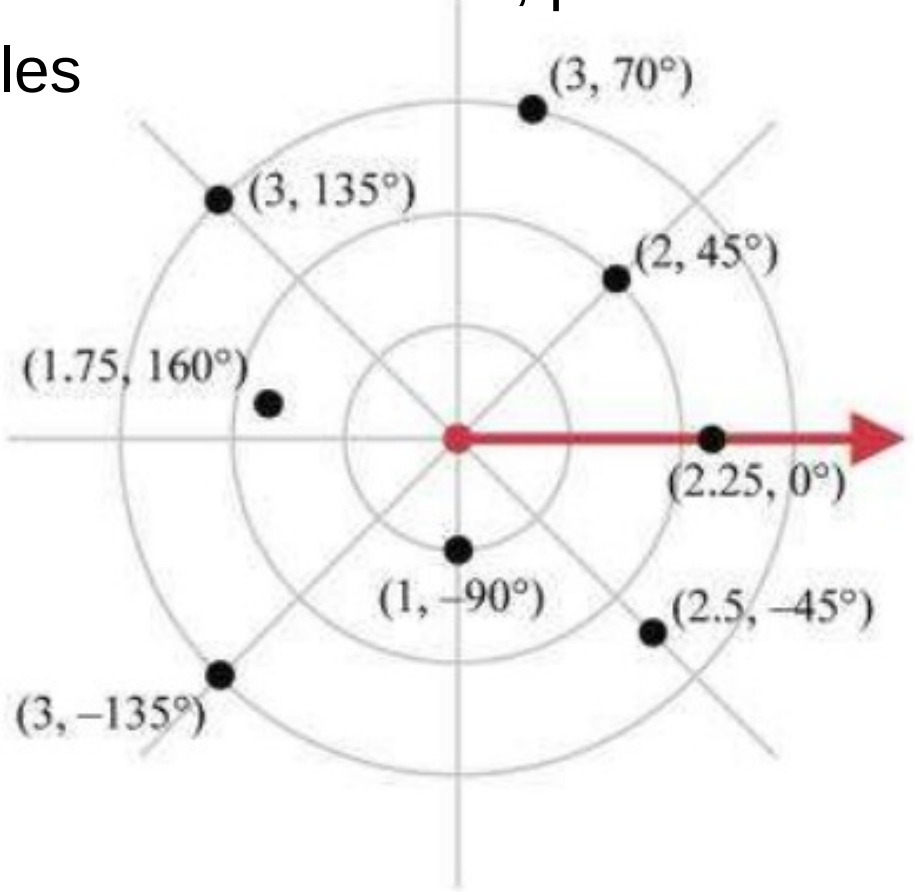
Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Représentation d'un point
    - Coordonnées cartésiennes : 2 distances signées (x,y)
    - Coordonnées polaires : une distance r et un angle θ



Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Représentation d'un point (r,θ)
    - D'abord on fait la rotation, puis on s'occupe de la distance
    - Exemples



Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Attention aux proportions
    - Dans un système de coordonnées cartésiennes, une unité peut représenter n'importe quoi (mètres, miles, années-lumières...), l'échelle est la même pour x et y, donc les proportions restent inchangées
    - Dans un système de coordonnées polaires
      - Les deux composantes r et θ n'ont pas la même unité
      - L'utilisation d'unités de mesure différentes pour θ peut changer les proportions
      - Degrés ou radians ?
        - Les humains préfèrent les degrés (180°, 360°...)
        - Les machines préfèrent les radians (π radians, 2π radians...)
        - Le tout est de choisir une unité et de s'y tenir

Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Problème de l'aliasing
    - Quelques questions spécifiques à un espace polaire
      - Est-ce que *r* peut être négatif ? (peut-on revenir en arrière ?)
      - Est-ce que θ peut aller au-delà de l'intervalle [-180°,180°] ?
      - Quelle est la valeur de θ directement "à l'ouest" ? + ou – 180° ?
      - Quand r=0, quelle est la valeur de θ ?
    - Une même réponse : oui
      - Pour un point donné, il y a une infinité de couples de coordonnées polaires qui peuvent servir à le décrire
    - Une définition : plusieurs références pour une donnée
      - Deux paires de coordonnées polaires sont **alias** l'une de l'autre si elles ont différentes valeurs, mais décrivent le même point
      - Les alias du couple (r,θ) sont tous les couples $((-1)^k * r, \theta + k * 180°)$

Coordonnées polaires

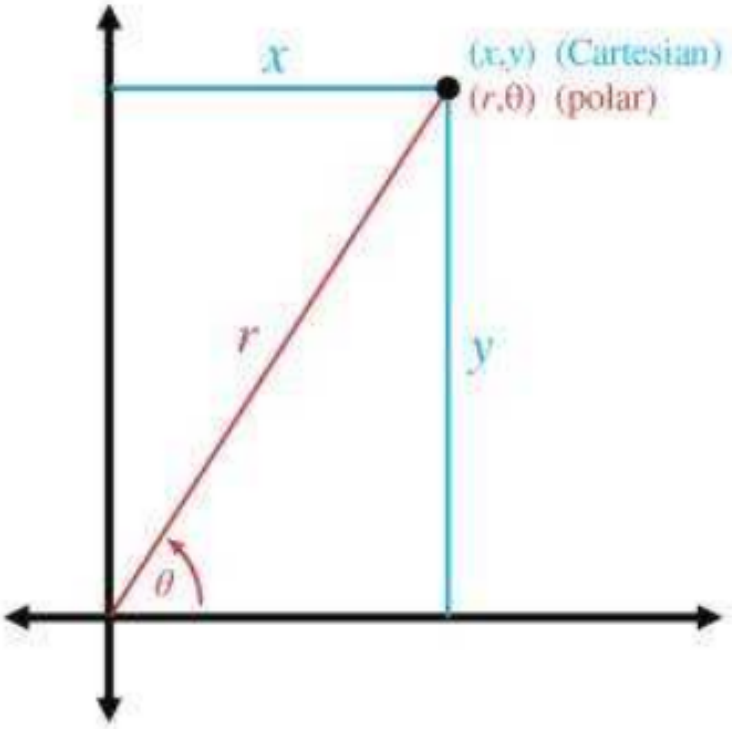
- Espace de coordonnées polaires en 2D
  - Problème de l'aliasing
    - Une solution : mettre les coordonnées en forme **canonique**
      - Propriétés
        - On ne mesure pas le retour en arrière : $r \geq 0$
        - Un angle est limité à une demi-révolution : $-180° < \theta \leq 180°$
        - A l'origine, l'angle est nul : $r = 0 \Rightarrow \theta = 0$
      - Algorithme correspondant
        - Si $r = 0$ alors $\theta \leftarrow 0$
        - Si $r < 0$ alors $r \leftarrow -r$, et on ajoute 180° à θ
        - Si $\theta \leq -180°$ alors on ajoute 360° à θ jusqu'à ce que $\theta > -180°$
        - Si $\theta > 180°$ alors on soustrait 360° à θ jusqu'à ce que $\theta \leq 180°$

Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Conversion entre coordonnées cartésiennes et coordonnées polaires
    - Coordonnées polaires (r,θ) → coordonnés cartésiennes

$x = r \cos \theta$

$y = r \sin \theta$



Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Conversion entre coordonnées cartésiennes et coordonnées polaires
    - Coordonnées cartésiennes (x,y) → coordonnées polaires
      - Pour trouver r

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

      - Pour trouver θ

$$\frac{y}{x} = \frac{r \sin\theta}{r \cos\theta},$$

$$\frac{y}{x} = \frac{\sin\theta}{\cos\theta},$$

$$y/x = \tan\theta,$$

$$\theta = \arctan(y/x)$$

Coordonnées polaires

- Espace de coordonnées polaires en 2D
  - Conversion entre coordonnées cartésiennes et coordonnées polaires
    - Problèmes
      - Si *x*=0, la division n'est pas possible
      - La fonction arctan est confinée dans l'intervalle [-90°,+90°]
      - Quand *x*/*y* >0, est-ce que *x*>0 et *y*>0 ? Ou *x*<0 et *y*<0 ?
    - Solution : la fonction `atan2`
    - Résultat

$$r = \sqrt{x^2 + y^2}$$
$$\theta = \operatorname{atan2}(y, x)$$

$$\operatorname{atan2}(y, x) = \begin{cases} 0, & x = 0, y = 0, \\ +90^\circ, & x = 0, y > 0, \\ -90^\circ, & x = 0, y < 0, \\ \arctan(y/x), & x > 0, \\ \arctan(y/x) + 180^\circ, & x < 0, y \geq 0, \\ \arctan(y/x) - 180^\circ, & x < 0, y < 0. \end{cases}$$

Coordonnées polaires

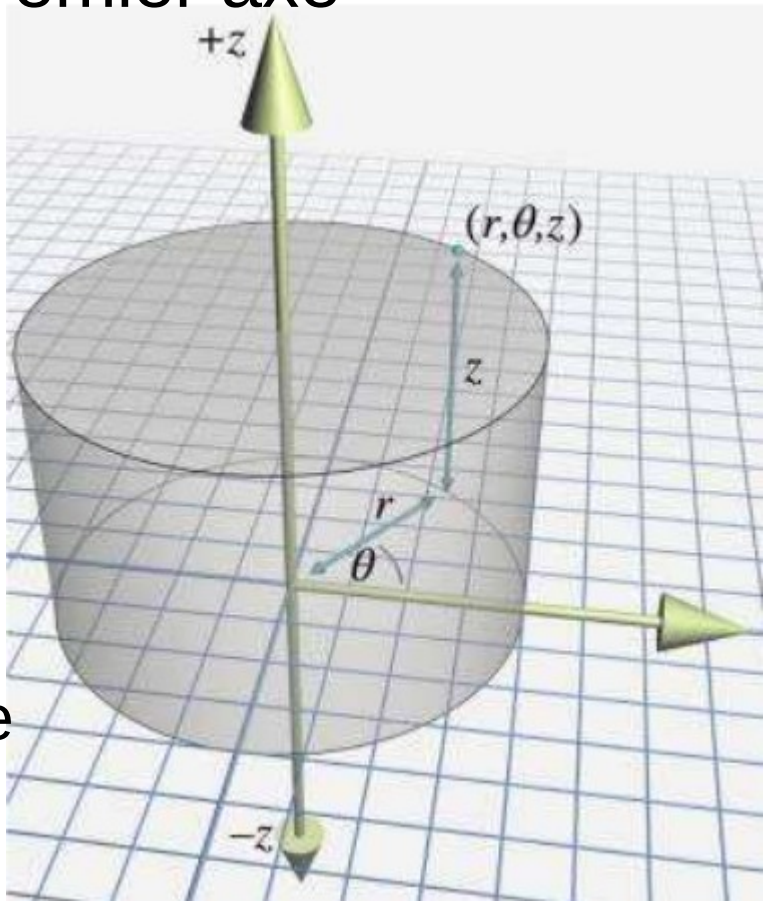
- A quoi servent les coordonnées polaires ?
  - Elles sont plus "naturelles" pour nous
    - Un humain ne décrit pas sa position avec des coordonnées purement cartésiennes
    - "Tourner à droite dans 5km", "10km au sud-ouest de..."
    - La terre et ronde. La latitude et la longitude sont polaires
  - Dans un monde 3D, on en a constamment besoin
    - Orienter la caméra (celle du joueur, celle du jeu...)
    - Rotation d'un modèle dans un outil ou un moteur 3D

Coordonnées polaires

- Espace de coordonnées polaires en 3D
  - La 3ème dimension rajoute une 3ème coordonnée
  - Cette 3ème coordonnée peut être
    - Une deuxième distance (comme r)
      - Dans ce cas, on parle de coordonnées **cylindriques**
      - Ces coordonnées sont plus intuitives
    - Un deuxième angle (comme θ)
      - Dans ce cas, on parle de coordonnées **sphériques**
      - Ces coordonnées sont les plus communes et les plus utilisées

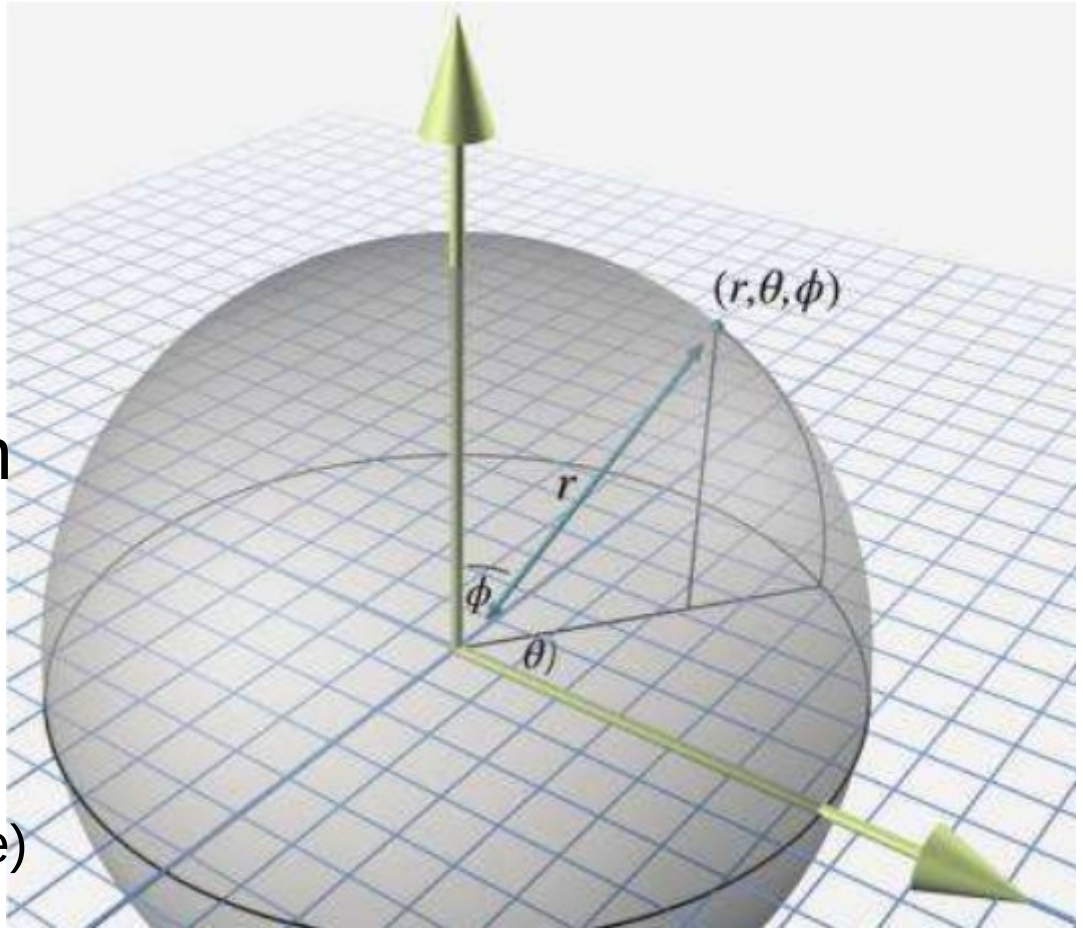
Coordonnées polaires

- Coordonnées 3D cylindriques
  - On rajoute une deuxième distance selon un axe (*z*) vertical et perpendiculaire au premier axe
  - Pour localiser un point
    - On cherche r et θ comme en 2D
    - On va en haut (ou en bas)
  - Pour convertir en coordonnées cartésiennes
    - On convertit r et θ comme en 2D
    - La conversion de z est immédiate

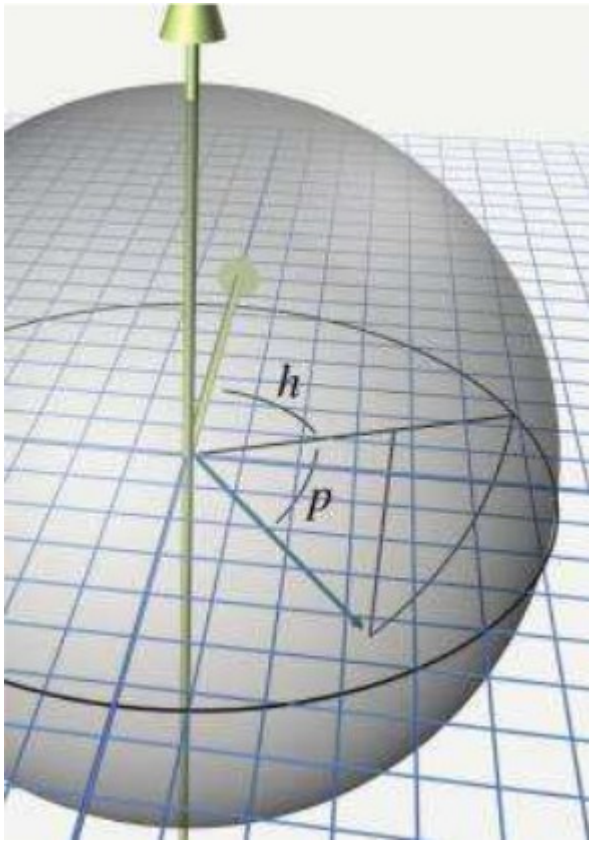


Coordonnées polaires

- Coordonnées 3D sphériques
  - Une distance **radiale** *r*
  - Deux axes polaires
    - Un horizontal (*x*)
    - Un vertical (*y*)
  - Deux angles qui définissent la direction
    - Par rapport à *x* : θ
    - Par rapport à *y* : Φ
    - Rotation positive
      - Antihoraire (main droite)
      - Horaire (main gauche)



Coordonnées polaires



- Coordonnées 3D sphériques
  - Vocabulaire et équivalences
    - L'angle Θ est appelé **azimuth**
      - C'est aussi la latitude
    - L'angle Φ est appelé **zénith**
      - La longitude correspond à 90°- Φ
  - Les conventions de notation varient
    - Parfois, *r* est noté ρ
    - Les angles Θ et Φ sont parfois inversés (ex: en physique)
    - En informatique, dans un système main gauche
      - L'angle horizontal est renommé *h* (**heading**) et pointe devant
      - L'angle vertical est renommé *p* ou (**pitch**) et pointe vers le bas
      - Le sens positif de rotation est horaire

Coordonnées polaires

- Coordonnées 3D sphériques
  - D'abord, on effectue la rotation horizontale, ensuite la rotation verticale, et enfin on se déplace de *r*
  - Problème de l'aliasing et de la singularité
    - Un alias de (*h*,*p*) peut être généré par (*h* ± 180°, 180°- *p*)
      - Exemple : au lieu de tourner à droite de 90° et se baisser de 45°, on peut tourner à gauche de 90°, et se baisser de 180-45 = 135°
    - Quand l'angle *p* est de ±90°, *h* perd toute signification
      - Exemple : au lieu de tourner à gauche ou à droite de 35°, 50°, 110°... puis de se basser de 90°, il suffit... de se baisser de 90°
      - C'est ce qu'on appelle le **blocage de cardan** ou **gimbal lock** (nous y reviendrons)
  - Il nous faut à nouveau des coordonnées canoniques

Coordonnées polaires

- Coordonnées 3D sphériques
  - Forme canonique pour un espace 3D sphérique
    - Pas de retour en arrière

      $r \geq 0$
    - L'angle horizontal est limité à une demi-révolution

      $-180° < h \leq 180°$
    - L'angle vertical est limité à un quart de révolution

      $-90° < p \leq 90°$
    - A l'origine, les deux angles sont nuls

      $r = 0 \Rightarrow h = p = 0$
    - Si on regarde tout en haut ou tout en bas, l'angle horizontal est nul

      $|p| = 90° \Rightarrow h = 0$

Coordonnées polaires

- Coordonnées 3D sphériques
  - Forme canonique pour un espace 3D sphérique
    - Algorithme
      - Si $r = 0$, alors $h \leftarrow 0$ et $p \leftarrow 0$
      - Si $r < 0$, alors $r \leftarrow -r$, $h \leftarrow h + 180°$ et $p \leftarrow -p$
      - Si $p < -90°$, alors on ajoute 360° à p jusqu'à ce que $p \geq -90°$
      - Si $p > 270°$, alors on retranche 360° à p jusqu'à ce que $p \leq 270°$
      - Si $p > 90°$, alors $h \leftarrow h + 180°$ et $p \leftarrow 180° - p$
      - Si $h \leq -180°$, alors on ajoute 360° jusqu'à ce que $h > -180°$
      - Si $h > 180°$, alors on retranche 360° à h jusqu'à ce que $h \leq 180°$

Coordonnées polaires

- Coordonnées 3D sphériques
  - Conversion entre coordonnées cartésiennes et coordonnées sphériques
    - Convention "main-droite"
      - Soit un point (r,θ,Φ) (coord. polaires)
      - On veut l'équivalent cartésien (x,y,z)
      - Soit d la distance horizontale entre le point et l'axe vertical

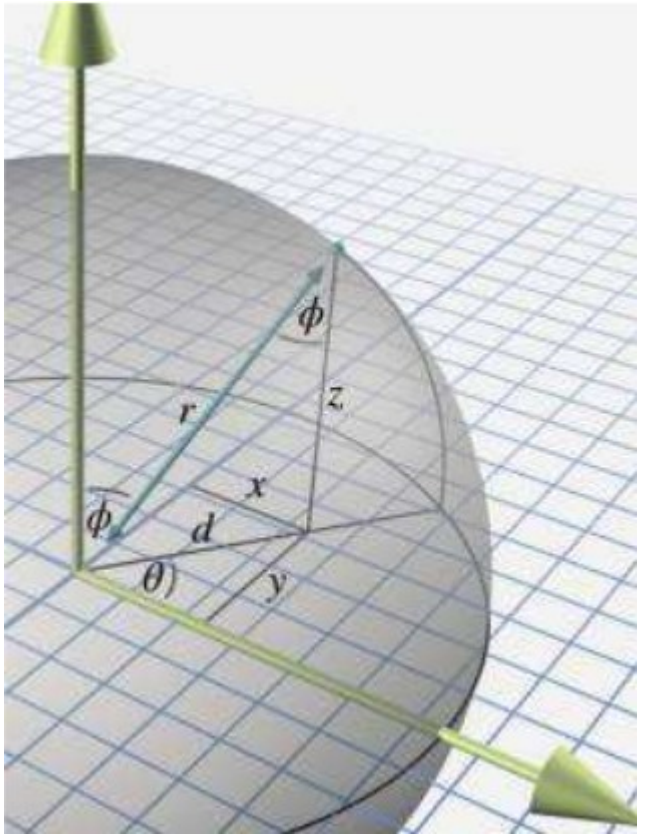
        $z / r = \cos \Phi$

        $d / r = \sin \Phi$

        $x / d = \cos \theta$

        $y / d = \sin \theta$
      - On en déduit

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



Coordonnées polaires

- Coordonnées 3D sphériques
  - Conversion entre coordonnées cartésiennes et coordonnées sphériques
    - Convention "main gauche" et notation informatique
      - Coordonnées sphériques → cartésiennes :

        on adapte les équations "main droite" précédentes :

        $x = r \cos p \sin h$ $y = -r \sin p$ $z = r \cos p \cos h$
      - Coordonnées cartésiennes → sphériques

$$r = \sqrt{x^2 + y^2 + z^2} \qquad h = \operatorname{atan2}(x, z)$$

$$y = -r \sin p,$$

$$-y/r = \sin p,$$

$$p = \arcsin(-y/r)$$

Coordonnées polaires

- Utilisation des coordonnées polaires pour spécifier des vecteurs
  - Les deux propriétés clés d'un vecteur sont la longueur et la direction
    - Sous forme polaire, elles sont décrites directement
    - Sous forme cartésienne, elles sont obtenues indirectement via des calculs impliquant une conversion en forme polaire
  - Les vecteurs sont aussi utilisés pour décrire des points (qui sont des "vecteurs position")
  - Les formules utilisées pour décrire les points d'une forme à l'autre marchent donc pour les vecteurs

Plan

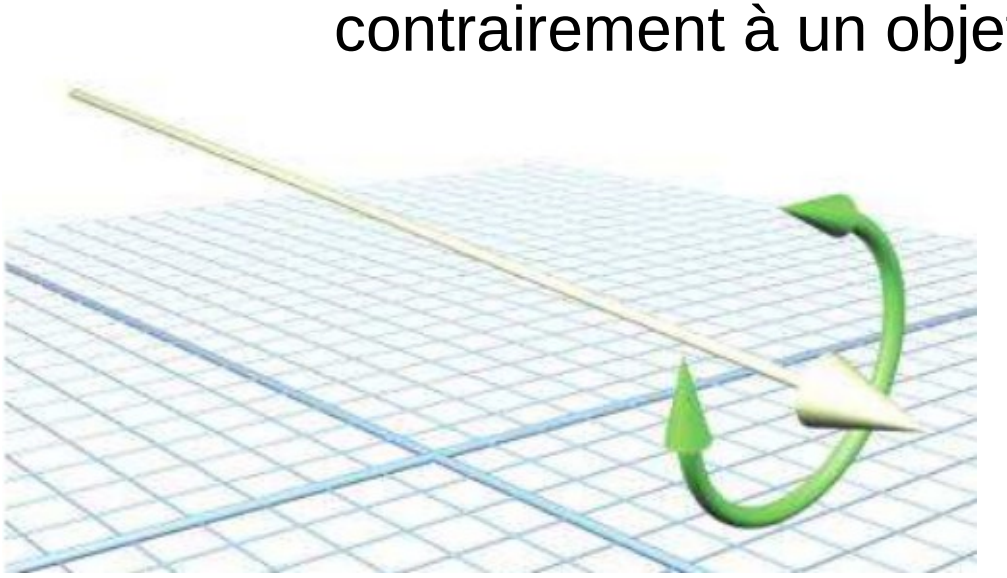
- Coordonnées polaires
- Rotations et orientation en 3D
  - Définitions
  - Matrices
  - Angles d'Euler
  - Quaternions
  - Comparaison entre les différentes méthodes
  - Conversion entre les représentations

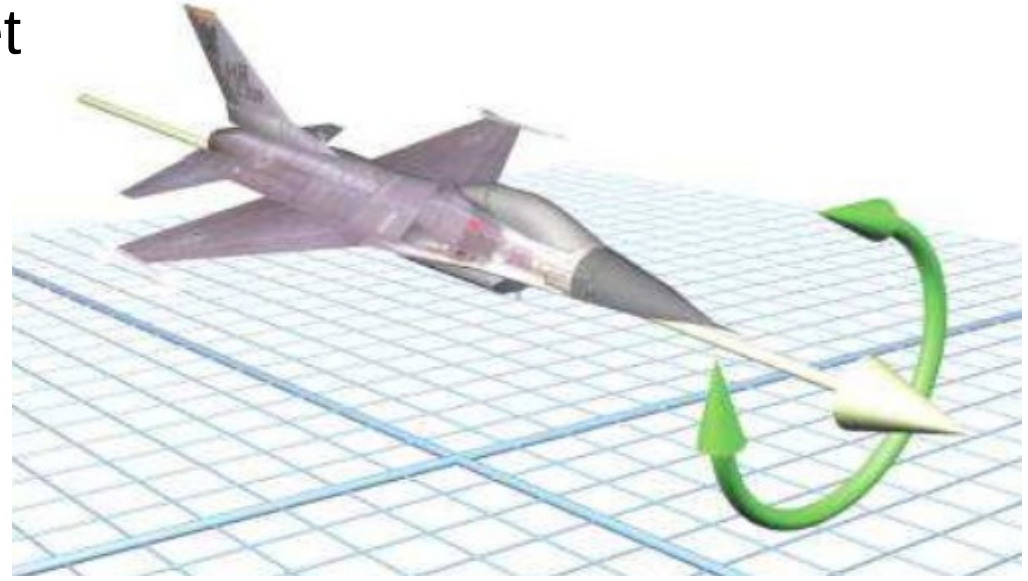
Rotations et orientation en 3D

- Décrire l'orientation d'un objet en 3D est un problème difficile
- Certaines notions ("orientation" et "déplacement angulaire") peuvent prêter à confusion
- Plusieurs méthodes existent, certaines plus adaptées que d'autres dans des cas précis
- Il est par conséquent nécessaire
  - De les connaître, avec leurs avantages, leurs limites
  - De savoir choisir la bonne méthode au bon moment
  - De savoir faire la conversion de l'une à l'autre

Rotations et orientation en 3D

- Définitions
  - Intuitivement, on sait que "l'orientation" d'un objet nous informe sur la direction vers laquelle il pointe
  - Cependant, "orientation" et "direction" ne veulent pas forcément dire la même chose
    - Exemple : quand on "tord" un vecteur, il ne change pas, contrairement à un objet





Rotations et orientation en 3D

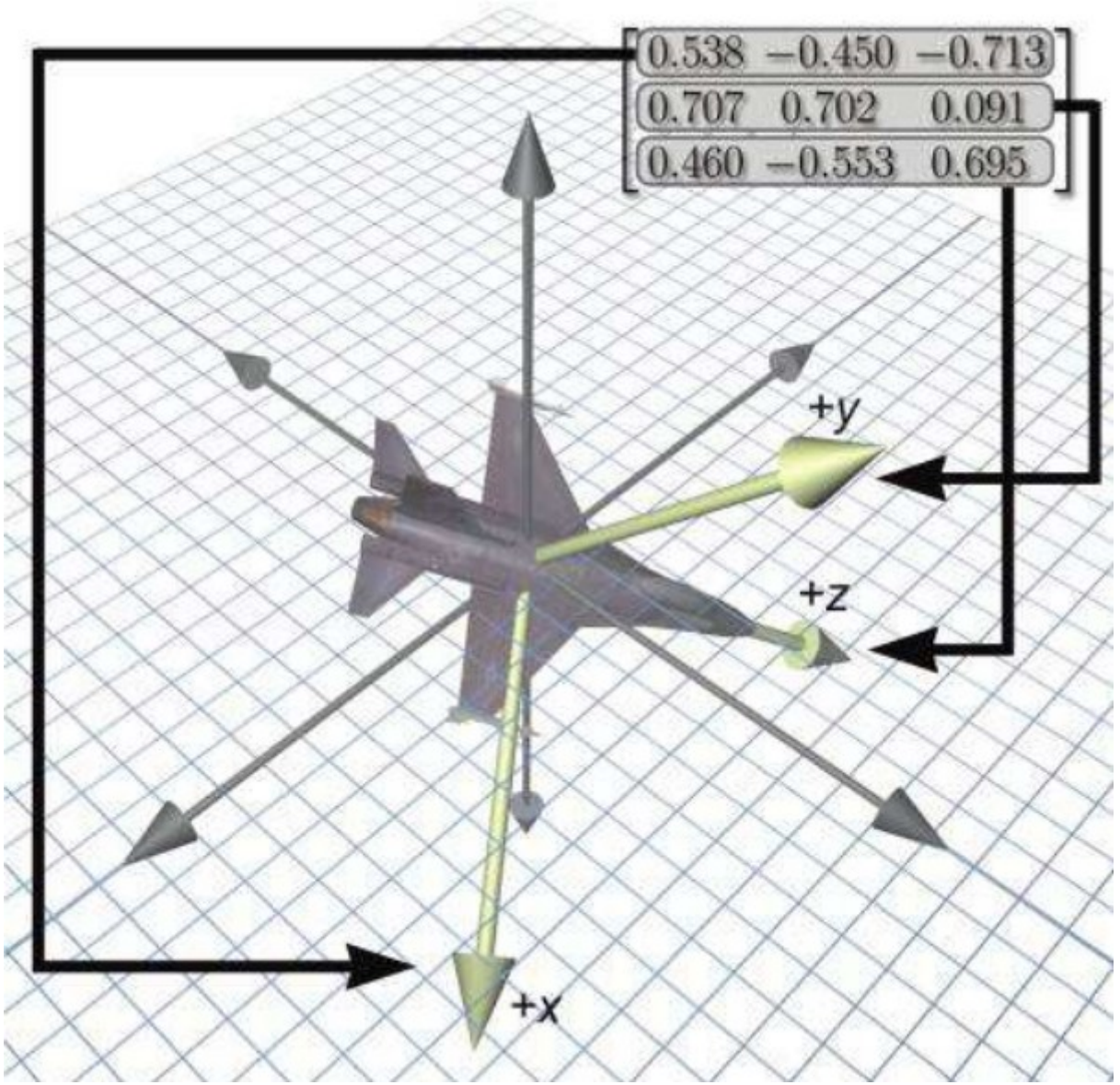
- Définitions
  - Différence de taille entre "direction" et "orientation"
    - On peut paramétrer une direction en 3D avec juste 2 nombres (angles des coordonnées sphériques)
      - Un vecteur a donc une direction, mais pas d'orientation
    - Pour l'orientation complète, on a besoin d'un 3ème angle
  - Notion de déplacement angulaire
    - Une rotation consiste à passer d'une orientation à une autre
    - Le degré de rotation est appelé "déplacement angulaire"
    - Décrire une orientation revient à décrire un déplacement angulaire... MAIS ce n'est pas la même chose

Rotations et orientation en 3D

- Définitions
  - Orientation vs. déplacement angulaire
    - Même distinction qu'entre points et vecteurs
      - Deux notions mathématiquement équivalentes, mais pas identiques conceptuellement
      - La première notion désigne un seul état, la deuxième désigne la différence entre deux états, la direction prise d'un état à l'autre
      - De même qu'un point peut être désigné par un "vecteur position" un déplacement angulaire peut désigner un "vecteur orientation"
    - Lien avec les représentations que nous allons voir
      - Les matrices et quaternions décrivent des déplacements angulaires
      - Les angles d'Euler, quant à eux, décrivent des orientations

Rotations et orientation en 3D

- Matrices
  - Avantages
    - Format utilisé par les API graphiques
    - Rotations immédiatement disponibles
    - Possibilité de concaténer les rotations (mult.)
    - Possibilité de défaire une rotation (inverse)



Rotations et orientation en 3D

- Matrices
  - Inconvénients d'une matrice
    - Encombrement de la mémoire
      - 9 nombres, alors que 3 sont vraiment nécessaires
    - Pas forcément intuitifs
      - A la base : uniquement des nombres, qu'il faut déchiffrer
    - Possibilité pour une matrice d'être "malformée"
      - Beaucoup de contraintes sur une "bonne" matrice de rotation
      - Exemple
        - On exécute plusieurs rotations les unes après les autres
        - Ce qui signifie une série de multiplications de matrices
        - Or il arrive qu'il y ait des erreurs de précision derrière la virgule
        - Avec une multiplication de matrices, ces erreurs risquent de s'accumuler jusqu'à devenir ingérables

Rotations et orientation en 3D

- Angles d'Euler
  - Nommés d'après le célèbre mathématicien qui les a développés, le suisse Leonhard Eu...

!!! PAUSE !!!

- Si vous voulez garder votre *street cred*...

  (surtout à l'étranger)

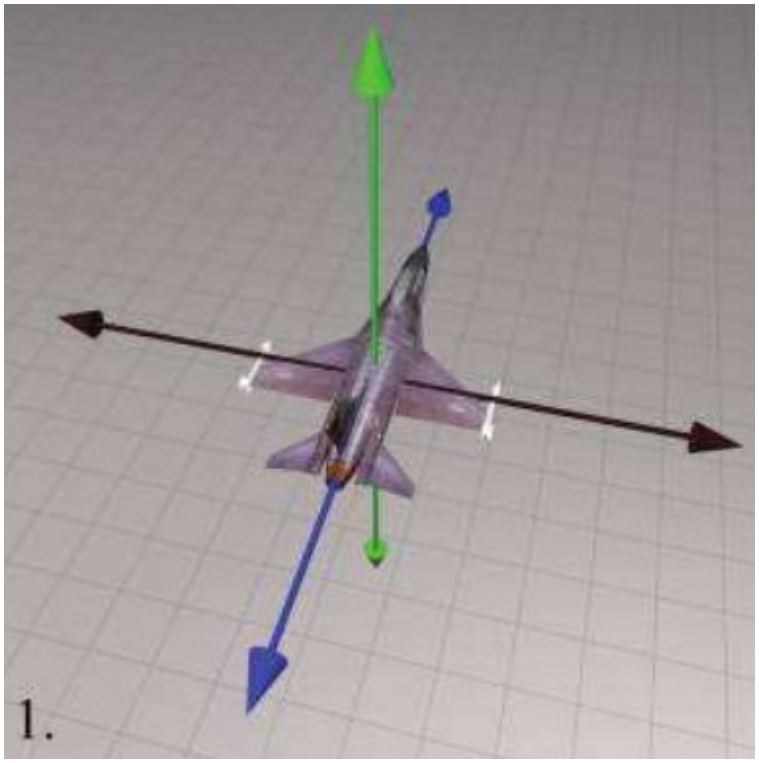
- ... vous ne dites pas "Euh-lère" ni "You-leure" mais **"Oy-leur"**

Rotations et orientation en 3D

- Angles d'Euler
  - Nommés d'après le célèbre mathématicien qui les a développés, le suisse Leonhard Euler (1707-1783)

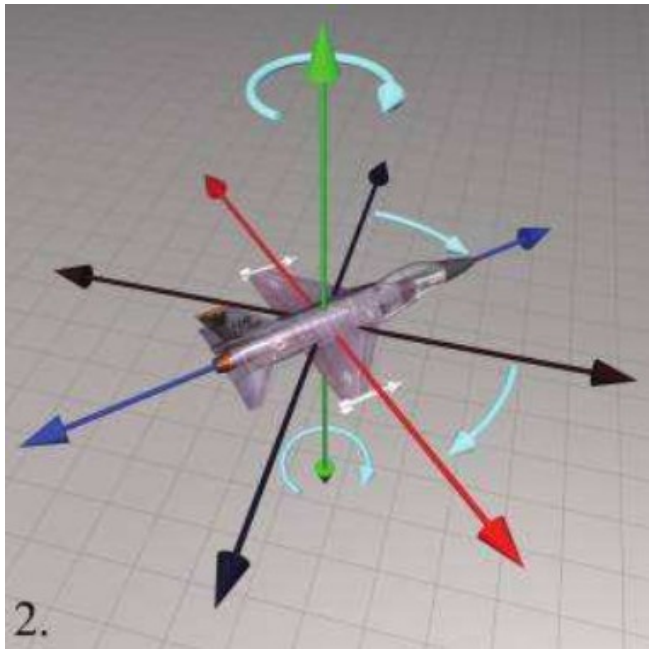
Rotations et orientation en 3D

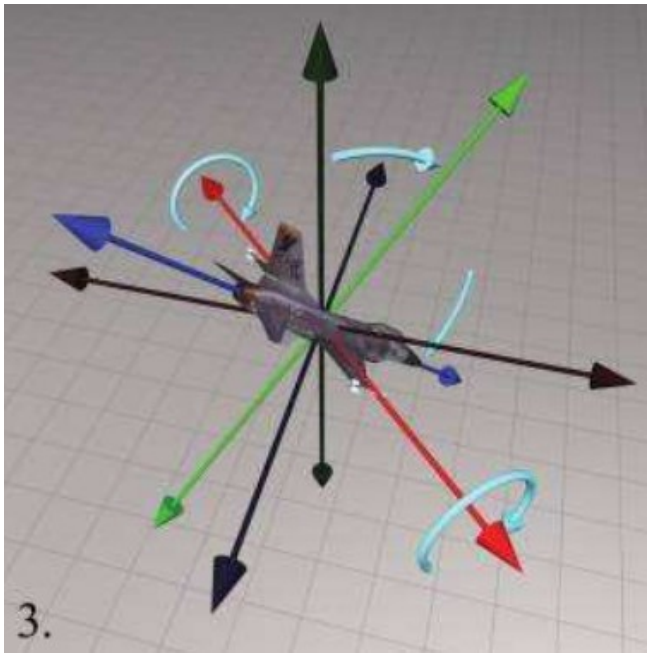
- Angles d'Euler
  - Idée de base
    - Un déplacement angulaire est défini comme une séquence de 3 rotations autour de 3 axes perpendiculaires 2 à 2
  - Confusion dans la spécification
    - On peut effectuer les rotations dans n'importe quel ordre
    - Il existe une douzaine (au moins) de notations possibles
  - Convention utilisée ici
    - Système "main gauche"
    - *x* à droite, *y* au-dessus, *z* au fond

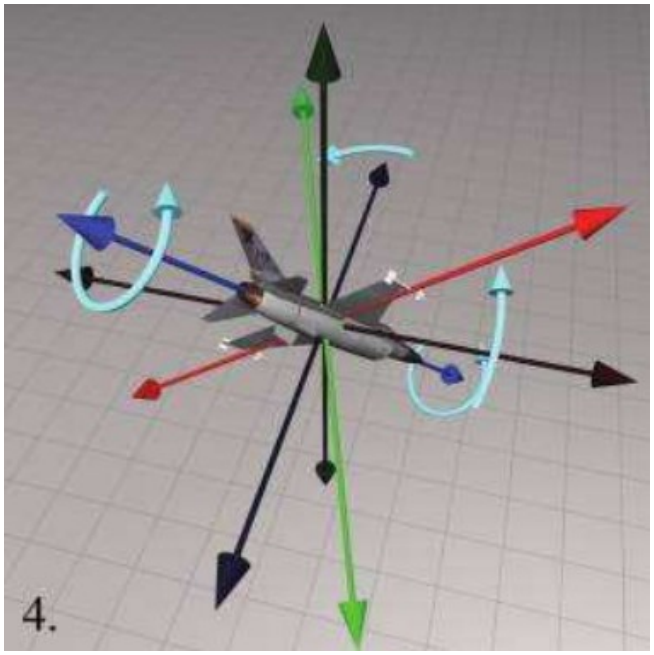


Rotations et orientation en 3D

- Angles d'Euler
  - Convention utilisée ici
    - D'abord, "heading" (axe *y*), sens positif horaire (à droite)
    - Ensuite, "pitch" (axe *x*), sens positif horaire (en bas)
    - Enfin, "bank" (axe *z*), sens positif antihoraire (à gauche)





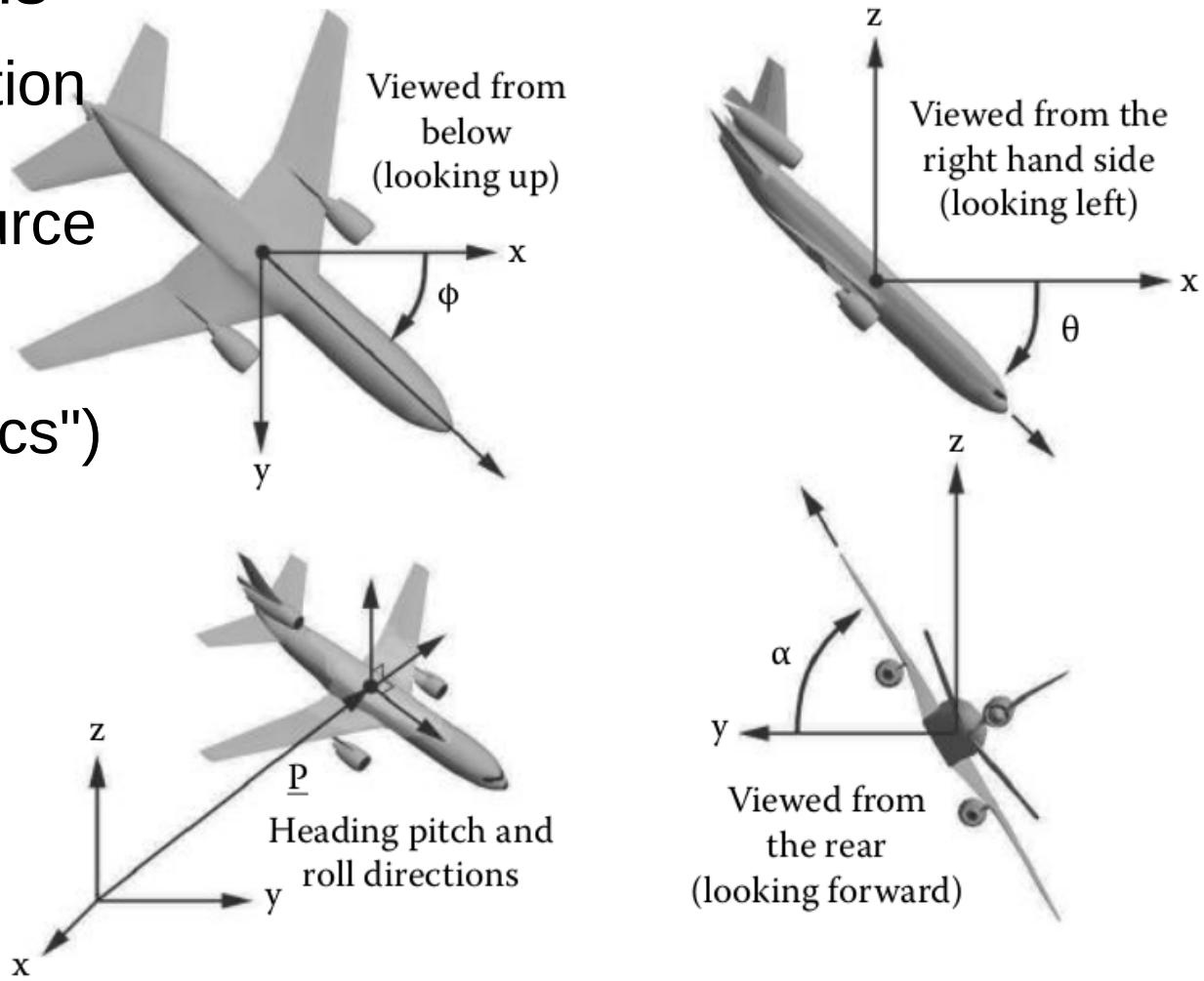


Rotations et orientation en 3D

- Angles d'Euler
  - Autres conventions
    - Noms de code : "yaw – pitch – roll"
      - Utilisée dans l'aéronautique ("tangage", "roulis", "lacet")
      - Héritée du vocabulaire marin (dans "aéronautique" il y a "nautique")
      - On la retrouve dans le livre "Game Engine Architecture"
      - Grosso modo, "yaw" ⇔ "heading" et "roll" ⇔ "bank"
      - Parfois, l'ordre change (exemple : notation "roll – pitch – yaw")
      - Autres expressions : "Azimuth – Elevation – Tilt" (ou "Twist")
    - Symboles mathématiques
      - (Φ,θ,α) (livre "Practical Algorithms for 3D Computer Graphics")
      - Autres notations : (θ,Φ,α), (Φ,θ,ψ), (ψ,θ,Φ), (Ω,i,ω), (α,β,Υ), ...
    - De façon générale
      - Attention à l'ordre dans lequel s'effectuent les rotations !

Rotations et orientation en 3D

- Angles d'Euler
  - Autres conventions
    - Exemple de notation pour un système "main droite" (source : "Practical Algorithms for 3D Computer Graphics")



Rotations et orientation en 3D

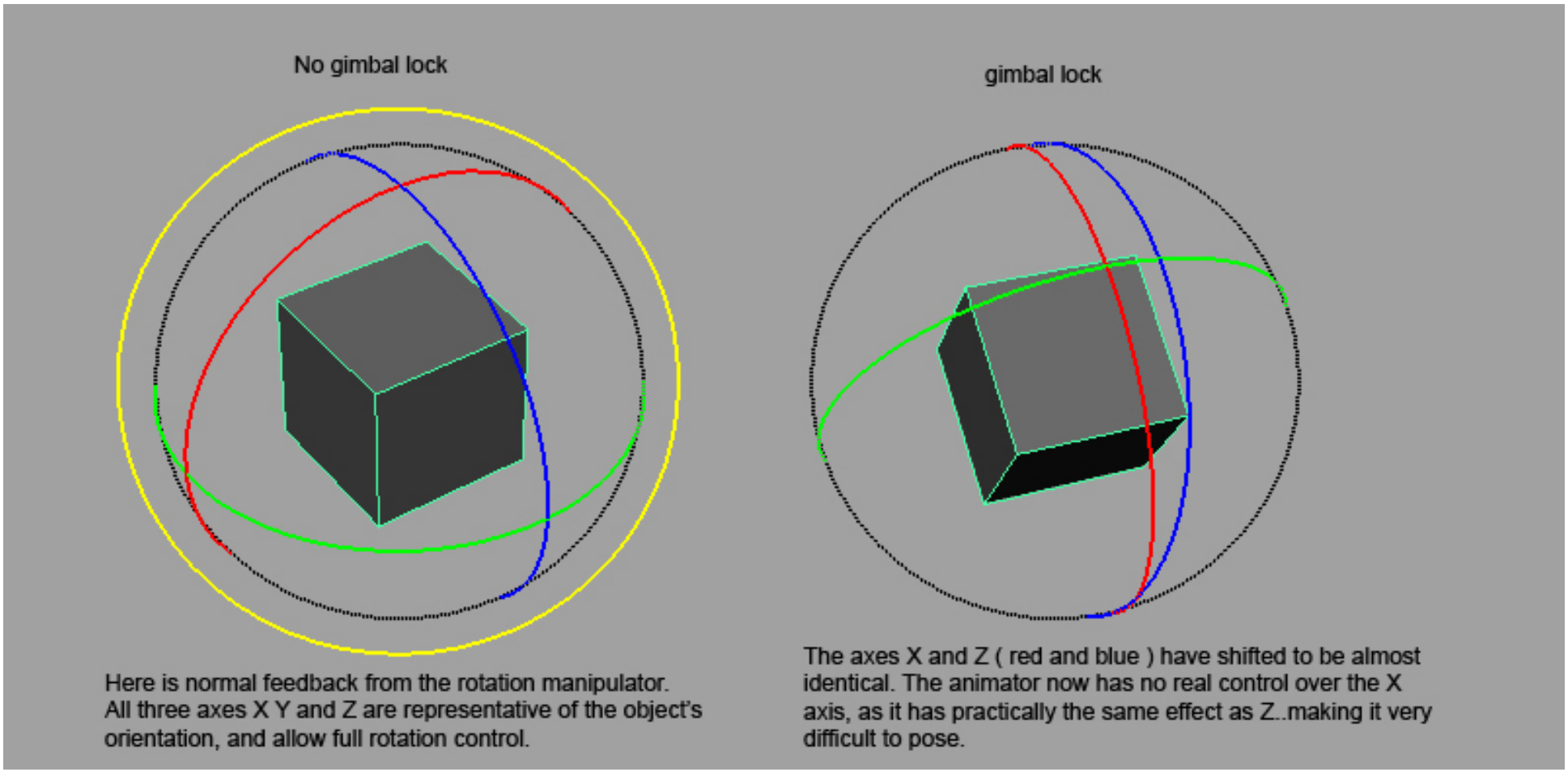
- Angles d'Euler
  - Avantages des angles d'Euler
    - Intuitifs et faciles d'utilisation du point de vue "humain"
    - La plus petite représentation possible d'une orientation
    - N'importe quel ensemble de 3 nombres est valide
  - Inconvénients des angles d'Euler
    - La représentation d'une orientation donnée n'est pas unique
    - L'interpolation entre deux orientations est problématique

Rotations et orientation en 3D

- Angles d'Euler
  - Où l'on retrouve le problème de l'aliasing
    - Pour une orientation donnée, il existe plusieurs triplets d'angles d'Euler capables de la décrire
    - Exemple : si on ajoute 360° à l'orientation, celle-ci ne change pas, mais les valeurs, si
    - Autre exemple : trois rotations heading 180°, puis pitch 45°, puis bank 180° équivalent à une rotation pitch 135°
  - Blocage de Cardan ("Gimbal lock")
    - Heading 45° puis pitch 90° ⇔ pitch 90°, puis bank 45°
    - Choisir un angle de ±90° pour pitch restreint fortement les deux autres rotations, qui se limitent autour de l'axe vertical

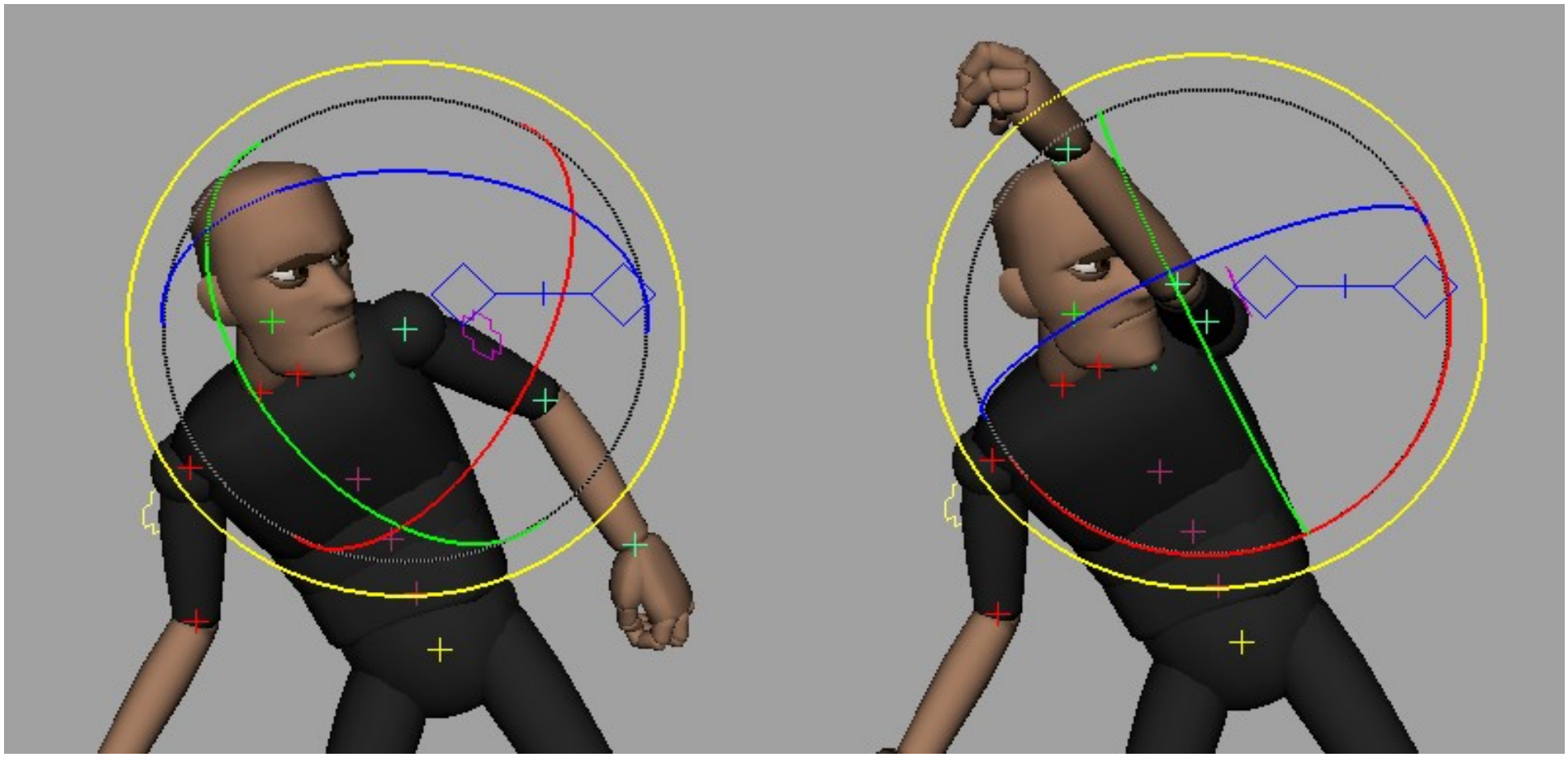
Rotations et orientation en 3D

- Angles d'Euler
  - Blocage de Cardan ("Gimbal lock")



Rotations et orientation en 3D

- Angles d'Euler
  - Blocage de Cardan ("Gimbal lock")



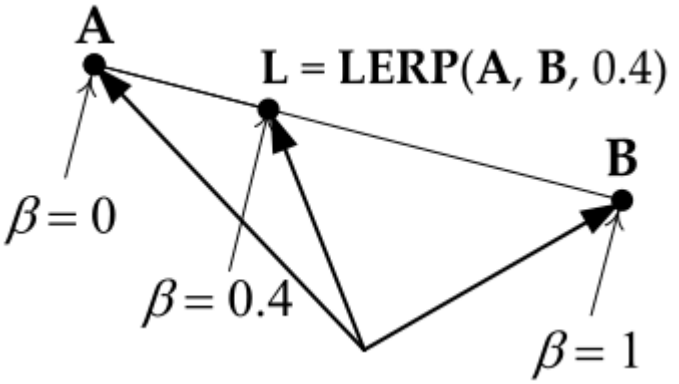
Rotations et orientation en 3D

- Angles d'Euler
  - Problème de l'interpolation
    - Interpolation : opération permettant de construire une courbe à partir d'un nombre fini de points
    - Interpolation linéaire (LERP) : permet de trouver un point intermédiaire entre deux points connus

      $\mathbf{l} = \text{LERP}(\mathbf{a},\mathbf{b},\beta) = (1 - \beta)\,\mathbf{a} + \beta\mathbf{b}$

      $= [(1 - \beta)\,a_x + \beta b_x, (1 - \beta)\,a_y + \beta b_y, (1 - \beta)\,a_z + \beta b_z]$
    - Interprétation géométrique

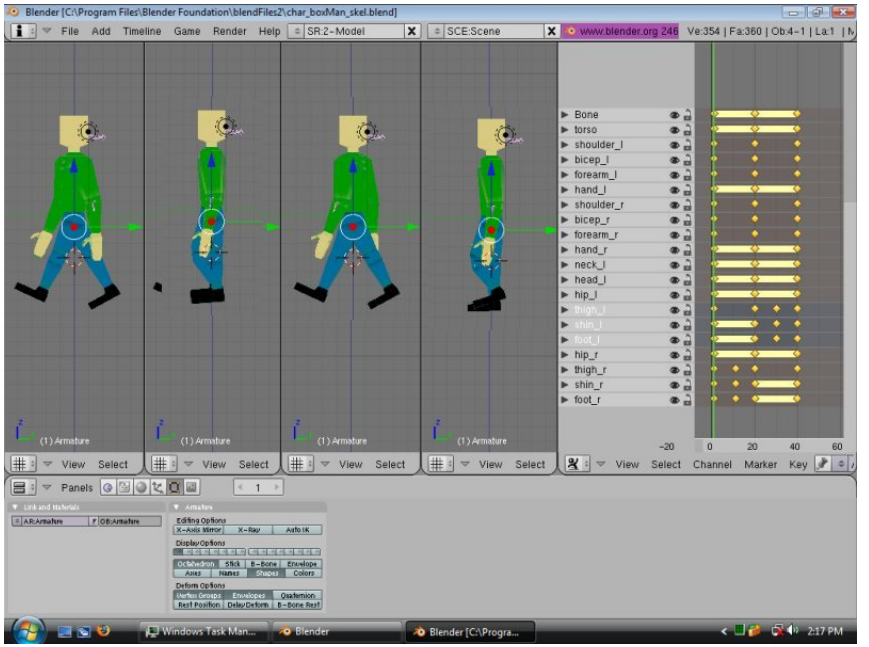
      LERP(**a**,**b**,β) est le vecteur position d'un point qui se trouve à β% du segment entre **a** et **b**



Rotations et orientation en 3D

- Angles d'Euler
  - Problème de l'interpolation
    - Indispensable pour l'animation et le contrôle de la caméra
      - Exemple : si on veut animer un objet d'un point **a** à un point **b** sur une séquence de 2 secondes à 30 FPS, il faut trouver 60 vecteurs positions intermédiaires entre **a** et **b**
      - Animation sous Blender

        (et autres logiciels 3D)
        - Simuler un "travelling" avec la caméra
        - Animation de personnage ("keyframes")
      - Editeur de personnage dans un RPG
      - Etc...

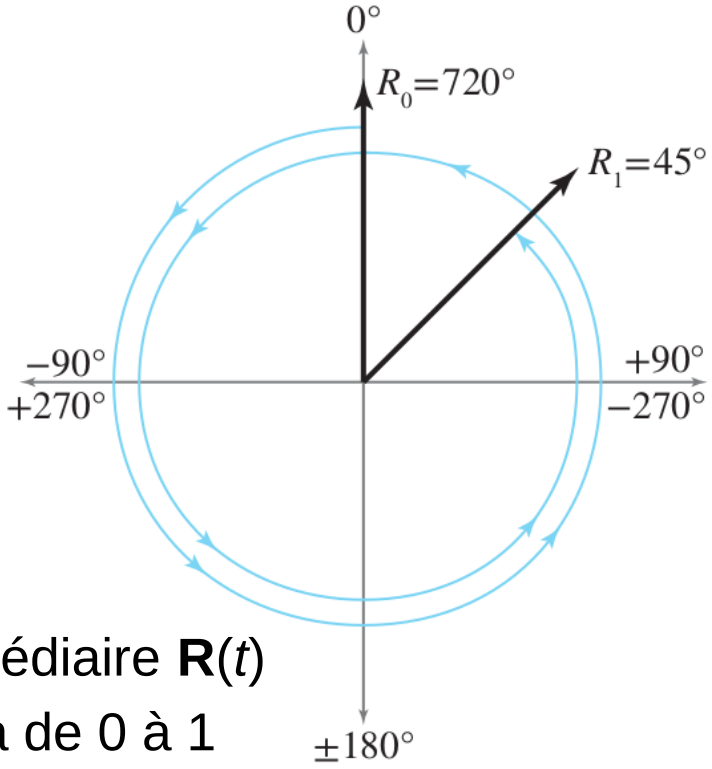


Rotations et orientation en 3D

- Angles d'Euler
  - Problème de l'interpolation
    - Interpolation angulaire
      - Soient deux orientations $\mathbf{R_0}$ et $\mathbf{R_1}$
      - Pour un paramètre t donné, $0 \leq t \leq 1$,
        on veut calculer une orientation intermédiaire $\mathbf{R}(t)$
        qui interpole de $\mathbf{R_0}$ à $\mathbf{R_1}$ tandis que t va de 0 à 1
      - Approche "naïve"
        - Interpolation linéaire (LERP) entre 2 angles :
        - On l'applique sur les 3 angles d'Euler

$$\Delta\theta = \theta_1 - \theta_0,$$
$$\theta_t = \theta_0 + t\,\Delta\theta$$

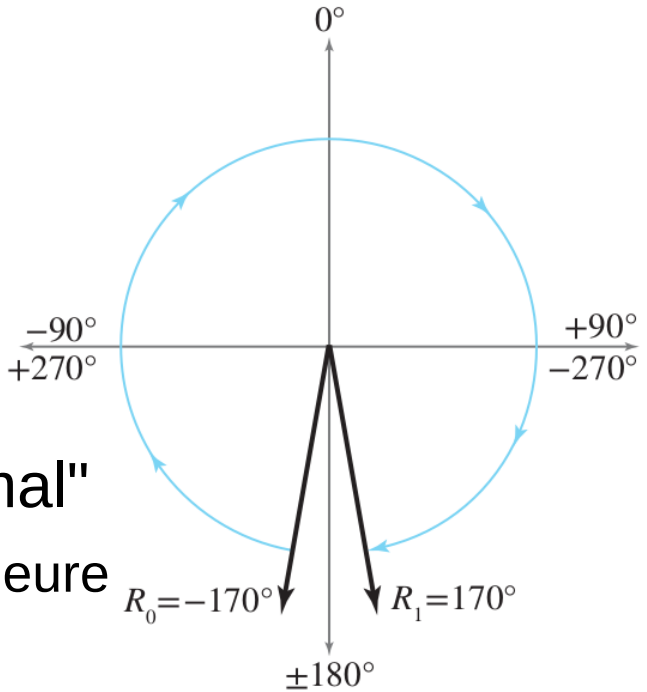
      - Problème
        - Soient h_0 et h_1 les angles h de $\mathbf{R_0}$ et $\mathbf{R_1}$, avec h_0=720° et h_1=45°
        - h_0 et h_1 sont éloignés de 45° seulement
        - Or, avec LERP, on tourne 2 fois dans la mauvaise direction



Rotations et orientation en 3D

- Angles d'Euler
  - Problème de l'interpolation
    - Forme canonique
      - On limite la rotation des 3 angles
      - Si pitch est de ±90°, bank est mis à 0
    - Encore un problème malgré tout
      - Supposons que h_0 = -170° et h_1 = +170°
      - Ce sont des angles canoniques
      - Ils ne sont séparés que de 20°
      - Et pourtant, avec LERP, on fait une rotation horaire de 340°
    - Toujours pas de chemin "torque minimal"
      - Chemin le plus direct sur la surface extérieure d'une sphère ("torque" = "collier")

$$-180^\circ < h \leq 180^\circ$$
$$-90^\circ \leq p \leq 90^\circ$$
$$-180^\circ < b \leq 180^\circ$$
$$p = \pm 90^\circ \quad \Rightarrow \quad b = 0$$



Rotations et orientation en 3D

- Angles d'Euler
  - Problème de l'interpolation
    - Solution : "Envelopper" les angles de rotation entre ±180°

$$\mathrm{wrapPi}(x) = x - 360^\circ \lfloor (x + 180^\circ)/360^\circ \rfloor$$

(signifie qu'on renvoie l'entier le plus proche de *a*)

    - L'interpolation devient alors : $\Delta\theta = \mathrm{wrapPi}(\theta_1 - \theta_0),$

$$\theta_t = \theta_0 + t\,\Delta\theta.$$

    - Toujours un problème !
      - Certes, on n'a plus d'aliasing...
      - Certes, on a une interpolation "torque minimale"...
      - MAIS on a encore un risque de blocage de Cardan
      - ... et le pire, c'est qu'on ne peut pas le résoudre complètement
      - En effet, c'est un problème inhérent à cette représentation
        - On n'a "que" 3 nombres pour représenter un espace 3D
    - Solution : les quaternions

Rotations et orientation en 3D

- Quaternions
  - Motivations
    - On a vu que pour représenter les translations en 3D, on passait par des matrices (et vecteurs) 4D
    - Pour les rotations, c'est le même principe
      - 3 valeurs pour 3 dimensions ⇒ problèmes de singularité
      - Dans ce cas, on va prendre 4 valeurs !
  - Utilisation
    - Depuis longtemps dans les jeux vidéo
      - cf. *Tomb Raider* (1996)
    - Omniprésent dans les moteurs 3D
      - Unity, Unreal Engine, Blender...

Rotations et orientation en 3D

- Quaternions
  - Développés par Sir William Rowan Hamilton en 1843 comme extension des nombres complexes
    - De fait, on peut les interpréter comme des nombres complexes 4D
  - Notation utilisée ici
    - Un quaternion contient deux composantes principales
      - Un scalaire w
      - Un vecteur $\mathbf{v}$ (ou $[x \;\; y \;\; z]$)
      - La notation est alors $[w \;\; \mathbf{v}]$ ou $[w \;\; (x \;\; y \;\; z)]$
      - On peut aussi le noter verticalement (ici, ça ne change rien)
      - Notation complexe : $\mathbf{q} = w + ix + jy + kz$

William Rowan Hamilton
(1805-1865)

Rotations et orientation en 3D

- Quaternions
  - Interprétation géométrique
    - Soit θ l'angle de rotation
    - Soit $\hat{\mathbf{n}}$ le vecteur unitaire parallèle à l'axe de rotation
    - Alors le quaternion peut être défini comme suit

$$\begin{bmatrix} w & \mathbf{v} \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) & \sin(\theta/2)\hat{\mathbf{n}} \end{bmatrix}$$

    - Notation alternative

$$\begin{bmatrix} w & (x & y & z) \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) & (\sin(\theta/2)n_x & \sin(\theta/2)n_y & \sin(\theta/2)n_z) \end{bmatrix}$$

Rotations et orientation en 3D

- Quaternions
  - Négation d'un quaternion
    - On effectue la négation de toutes les composantes

$$-\mathbf{q} = -\begin{bmatrix} w & (x & y & z) \end{bmatrix} = \begin{bmatrix} -w & (-x & -y & -z) \end{bmatrix}$$
$$= -\begin{bmatrix} w & \mathbf{v} \end{bmatrix} = \begin{bmatrix} -w & -\mathbf{v} \end{bmatrix}.$$

    - Mais ça ne change en rien le déplacement angulaire
      - Explication
        - Reprenons la formule précédente

$$\begin{bmatrix} w & \mathbf{v} \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) & \sin(\theta/2)\hat{\mathbf{n}} \end{bmatrix}$$

        - Si on ajoute 360° à θ, ça ne change rien à la rotation
        - En revanche

$$\cos(\theta/2 + 180) = -\cos(\theta/2)$$
$$\sin(\theta/2 + 180) = -\sin(\theta/2)$$

Rotations et orientation en 3D

- Quaternions
  - Quaternion identité
    - Point de vue géométrique
      - Il existe deux quaternions "identité" qui signifient : "pas de déplacement angulaire"

        [1 **0**] et [-1 **0**]
    - Point de vue algébrique
      - Il n'existe en fait qu'un seul quaternion identité : [1 **0**]
      - Explication
        - Si on multiplie un quaternion **q** par [1 **0**], ça donne **q**
        - Si on multiplie un quaternion **q** par [-1 **0**], ça donne **-q**
        - Si géométriquement parlant, **q** et **-q** désignent le même déplacement angulaire, algébriquement parlant, ils ne sont pas égaux
        - Donc [**-1** **0**] n'est pas un "vrai" quaternion identité

Rotations et orientation en 3D

- Quaternions
  - Magnitude (norme) d'un quaternion

$$\|\mathbf{q}\| = \left\|\begin{bmatrix} w & (x & y & z)\end{bmatrix}\right\| = \sqrt{w^2 + x^2 + y^2 + z^2}$$
$$= \left\|\begin{bmatrix} w & \mathbf{v}\end{bmatrix}\right\| = \sqrt{w^2 + \|\mathbf{v}\|^2}.$$

  - Interprétation géométrique
    - Les quaternions de rotation sont unitaires (norme 1)
    - On parlera donc de **quaternions unitaires** pour les désigner

$$\|\mathbf{q}\| = \left\|\begin{bmatrix} w & \mathbf{v}\end{bmatrix}\right\| = \sqrt{w^2 + \|\mathbf{v}\|^2}$$
$$= \sqrt{\cos^2(\theta/2) + (\sin(\theta/2)\|\hat{\mathbf{n}}\|)^2}$$
$$= \sqrt{\cos^2(\theta/2) + \sin^2(\theta/2)\|\hat{\mathbf{n}}\|^2}$$
$$= \sqrt{\cos^2(\theta/2) + \sin^2(\theta/2)(1)}$$
$$= \sqrt{1}$$
$$= 1.$$

Rotations et orientation en 3D

- Quaternions
  - Conjugué et inverse d'un quaternion
    - Conjugué (**q***): négation de la partie vectorielle

$$\begin{aligned}\mathbf{q}^* &= \begin{bmatrix} w & \mathbf{v} \end{bmatrix}^* = \begin{bmatrix} w & -\mathbf{v} \end{bmatrix} \\ &= \begin{bmatrix} w & \begin{pmatrix} x & y & z \end{pmatrix} \end{bmatrix}^* = \begin{bmatrix} w & \begin{pmatrix} -x & -y & -z \end{pmatrix} \end{bmatrix}\end{aligned}$$

    - Inverse (**q**$^{-1}$): conjugué divisé par la norme $\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|}$
  - Interprétation géométrique
    - Pour un quaternion unitaire, la norme est 1, donc le conjugué et l'inverse sont équivalents
    - Le conjugué (ou l'inverse) d'un quaternion unitaire représente le déplacement angulaire opposé
      - D'ailleurs, quand on multiplie un quaternion par son inverse, on obtient le quaternion identité [1 **0**]

Rotations et orientation en 3D

- Quaternions
  - Addition de quaternions
    - On additionne les composantes une à une

$$\mathbf{q_1}+\mathbf{q_2} = [w_1 \;\; \mathbf{v_1}] + [w_2 \;\; \mathbf{v_2}] = [w_1+w_2 \;\; \mathbf{v_1}+\mathbf{v_2}]$$
$$= [w_1+w_2 \;\; (x_1+x_2 \;\; y_1+y_2 \;\; z_1+z_2)]$$

  - Produit de quaternions (ou "produit de Hamilton")

$$\mathbf{q}_1\mathbf{q}_2 = \begin{bmatrix} w_1 & (x_1 & y_1 & z_1) \end{bmatrix} \begin{bmatrix} w_2 & (x_2 & y_2 & z_2) \end{bmatrix}$$

$$= \begin{bmatrix} w_1w_2 - x_1x_2 - y_1y_2 - z_1z_2 \\ \begin{pmatrix} w_1x_2 + x_1w_2 + y_1z_2 - z_1y_2 \\ w_1y_2 + y_1w_2 + z_1x_2 - x_1z_2 \\ w_1z_2 + z_1w_2 + x_1y_2 - y_1x_2 \end{pmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} w_1 & \mathbf{v}_1 \end{bmatrix} \begin{bmatrix} w_2 & \mathbf{v}_2 \end{bmatrix}$$

$$= \begin{bmatrix} w_1w_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 & w_1\mathbf{v}_2 + w_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2 \end{bmatrix}$$

Rotations et orientation en 3D

- Quaternions
  - Produit de quaternions (ou "produit de Hamilton")
    - Propriétés
      - Similaire au produit vectoriel (moins la croix)
        - Renvoie un quaternion, et n'est pas commutatif
      - Associatif, mais pas commutatif

$$(\mathbf{ab})\mathbf{c} = \mathbf{a}(\mathbf{bc})$$

$$\mathbf{ab} \neq \mathbf{ba}.$$

      - Magnitude du produit = produit des magnitudes

$$\|\mathbf{q}_1\mathbf{q}_2\| = \|\mathbf{q}_1\|\|\mathbf{q}_2\|$$

      - Inverse du produit = produit des inverses dans l'ordre inverse

$$(\mathbf{ab})^{-1} = \mathbf{b}^{-1}\mathbf{a}^{-1},$$

$$(\mathbf{q}_1\mathbf{q}_2\cdots\mathbf{q}_{n-1}\mathbf{q}_n)^{-1} = {\mathbf{q}_n}^{-1}{\mathbf{q}_{n-1}}^{-1}\cdots{\mathbf{q}_2}^{-1}{\mathbf{q}_1}^{-1}$$

Rotations et orientation en 3D

- Quaternions
  - Produit de quaternions (ou "produit de Hamilton")
    - Utilisation pour la rotation d'un vecteur 3D
      - Soit un point 3D (x,y,z) (ou plutôt un vecteur position [x y z]
      - Étendons-le pour en faire un quaternion **p** = [0 (x y z)]
      - Soit q un quaternion unitaire **q** = [cos $\theta/2$, $\hat{\mathbf{n}}$ sin $\theta/2$]
      - La rotation **p'** de **p** autour de θ s'effectue comme suit : $\mathbf{p'}=\mathbf{qpq}^{-1}$
    - Concaténation de plusieurs rotations
      - Soient les quaternions unitaires **a** et **b**
      - La rotation de **p** par **a**, puis par **b**, s'effectue comme suit :

$$\begin{aligned}\mathbf{p}' &= \mathbf{b}(\mathbf{apa}^{-1})\mathbf{b}^{-1} \\ &= (\mathbf{ba})\mathbf{p}(\mathbf{a}^{-1}\mathbf{b}^{-1}) \\ &= (\mathbf{ba})\mathbf{p}(\mathbf{ba})^{-1}.\end{aligned}$$

      - Une rotation par **a** puis par **b** équivaut à une unique rotation par le produit de quaternions **ba**

Rotations et orientation en 3D

- Quaternions
  - "Différence" entre deux quaternions
    - Soient deux orientations **a** et **b**
    - On peut calculer le déplacement angulaire **d** qui effectue la rotation de **a** vers **b**
      - On appelle ça "différence", mais c'est plus une division
    - Le déplacement s'effectue comme suit : **da** = **b**
      - Le produit de quaternions exécute les rotations de droite à gauche
    - On va à présent calculer **d** :

$$
\begin{aligned}
(\mathbf{da})\mathbf{a}^{-1} &= \mathbf{ba}^{-1} \\
\mathbf{d}(\mathbf{aa}^{-1}) &= \mathbf{ba}^{-1} \\
\mathbf{d}\begin{bmatrix}1 & \mathbf{0}\end{bmatrix} &= \mathbf{ba}^{-1} \\
\mathbf{d} &= \mathbf{ba}^{-1}
\end{aligned}
$$

Rotations et orientation en 3D

- Quaternions
  - Produit scalaire de quaternions
    - Similaire au produit scalaire entre deux vecteurs
      - Le résultat est un scalaire
      - Il sert à mesurer la similarité entre deux orientations
    - Résultat du produit scalaire

$$\begin{aligned}\mathbf{q}_1 \cdot \mathbf{q}_2 &= \begin{bmatrix} w_1 & \mathbf{v}_1 \end{bmatrix} \cdot \begin{bmatrix} w_2 & \mathbf{v}_2 \end{bmatrix} \\ &= w_1 w_2 + \mathbf{v}_1 \cdot \mathbf{v}_2 \\ &= \begin{bmatrix} w_1 & \begin{pmatrix} x_1 & y_1 & z_1 \end{pmatrix} \end{bmatrix} \cdot \begin{bmatrix} w_2 & \begin{pmatrix} x_2 & y_2 & z_2 \end{pmatrix} \end{bmatrix} \\ &= w_1 w_2 + x_1 x_2 + y_1 y_2 + z_1 z_2.\end{aligned}$$

    - Pour un quaternion unitaire : $-1 \leq \mathbf{a} \cdot \mathbf{b} \leq 1$

Rotations et orientation en 3D

- Quaternions
  - Logarithme d'un quaternion
    - Soit **q** = [cos *θ*/2, **n̂** sin *θ*/2] un quaternion unitaire
    - Soit le demi-angle $\alpha = \theta/2$, tel que **q** = [cos *α*, **n̂**sin *α*]
    - Logarithme de **q** : $\log \mathbf{q} = \log\left(\begin{bmatrix}\cos\alpha & \hat{\mathbf{n}}\sin\alpha\end{bmatrix}\right) \equiv \begin{bmatrix}0 & \alpha\hat{\mathbf{n}}\end{bmatrix}$
  - Exponentielle d'un quaternion
    - Soit **p** un quaternion de la forme p = [0 $\alpha\hat{\mathbf{n}}$] ($||\hat{\mathbf{n}}|| = 1$)
    - Exponentielle de **p** : $\exp \mathbf{p} = \exp\left(\begin{bmatrix}0 & \alpha\hat{\mathbf{n}}\end{bmatrix}\right) \equiv \begin{bmatrix}\cos\alpha & \hat{\mathbf{n}}\sin\alpha\end{bmatrix}$
      - L'exponentielle est l'inverse du logarithme : exp(log **q**) = **q**
  - Multiplication d'un quaternion par un scalaire
    - Soient un scalaire *k* et un quaternion **q**
    - Alors $k\mathbf{q} = k[w \ \mathbf{v}] = [kw \ k\mathbf{v}]$

Rotations et orientation en 3D

- Quaternions
  - Exponentiation d'un quaternion
    - Multiplie le quaternion par lui-même un certain nombre de fois (c'est la fonction puissance d'un quaternion)
    - Pour un quaternion **q** et un scalaire t : $\mathbf{q}^t = \exp(t \log \mathbf{q})$
    - Quand t varie de 0 à 1, $\mathbf{q}^t$ varie de [1 **0**] à **q**
      - C'est utile pour calculer une partie du déplacement angulaire représenté par **q** (c'est une fraction de t)
      - Exemple : $\mathbf{q}^{1/3}$ représente 1/3 du déplacement angulaire **q**
    - Quand $t>1$, $\mathbf{q}^t$ représente t fois le déplacement angulaire **q**
      - Exemple : si **q**=30° alors $\mathbf{q}^2$ est une rotation horaire de 60°
    - Quand $t<0$, $\mathbf{q}^t$ représente une rotation dans le sens inverse
      - Exemple : si **q**=30° alors $\mathbf{q}^{-1/3}$ est une rotation antihoraire de 10°

Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - C'est LA raison pour laquelle les quaternions sont autant utilisés dans les jeux et applications 3D
    - Reprenons la formule standard de l'interpolation linéaire LERP entre deux scalaires a_0 et a_1

$$\Delta a = a_1 - a_0,$$
$$\operatorname{lerp}(a_0, a_1, t) = a_0 + t\,\Delta a.$$

    - Cette formule consiste en 3 étapes
      - D'abord, on calcule la différence entre a_1 et a_0
      - Ensuite, on prend une fraction de cette différence
      - Enfin, on "ajuste" a_0 avec cette fraction de différence

Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - Appliquons le principe pour interpoler entre deux orientations $\mathbf{q}_0$ et $\mathbf{q}_1$
      - Calcul de la "différence" entre $\mathbf{q}_1$ et $\mathbf{q}_0$

        $\Delta\mathbf{q} = \mathbf{q}_1\mathbf{q}_0^{-1}$
      - Fraction de cette "différence"

        $(\Delta\mathbf{q})^t$
      - Ajustement de $\mathbf{q}_0$ avec cette fraction de "différence"

        $(\Delta\mathbf{q})^t\mathbf{q}_0$
    - On obtient l'équation suivante pour SLERP

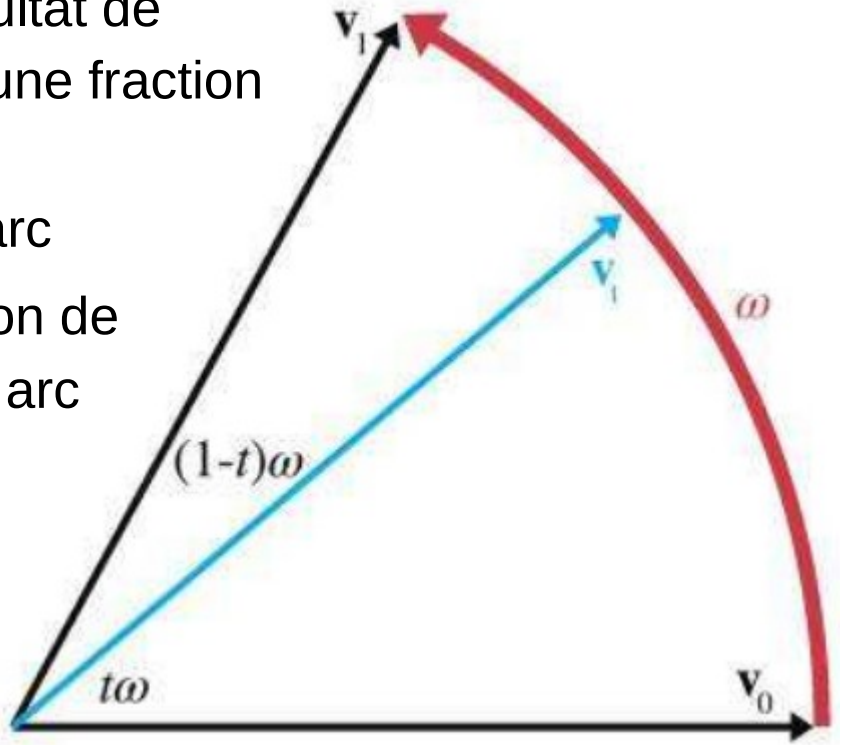
      $\text{slerp}(\mathbf{q}_0,\mathbf{q}_1,t) = (\mathbf{q}_1\mathbf{q}_0^{-1})^t\mathbf{q}_0$
    - Mais ce n'est que la forme algébrique, théorique

Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - Formule alternative (plus efficace)
      - Considérons l'espace 4D
      - Les quaternions qui nous intéressent étant unitaires, ils résident à la surface d'une hypersphère 4D
      - Nous allons donc interpoler autour de l'arc qui connecte les deux quaternions à la surface de cette hypersphère 4D
      - ... d'où la notion d'interpolation linéaire *sphérique*

Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - Formule alternative (plus efficace)
      - Soient deux vecteurs unitaires 2D $\mathbf{v}_0$ et $\mathbf{v}_1$
      - On veut calculer $\mathbf{v}_t$, qui est le résultat de l'interpolation autour de l'arc par une fraction t de la distance de $\mathbf{v}_0$ à $\mathbf{v}_1$
      - Soit ω l'angle entre $\mathbf{v}_0$ et $\mathbf{v}_1$ sur l'arc
      - Alors $\mathbf{v}_t$ est le résultat de la rotation de $\mathbf{v}_0$ d'un angle de $t\omega$ autour de cet arc



Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - Formule alternative (plus efficace)
      - Il existe des constantes positives k_0 et k_1 telles que $\mathbf{v_t} = k_0\mathbf{v_0} + k_1\mathbf{v_1}$ (combinaison linéaire)
      - Sachant que $\mathbf{v_1}$ est un vecteur unitaire

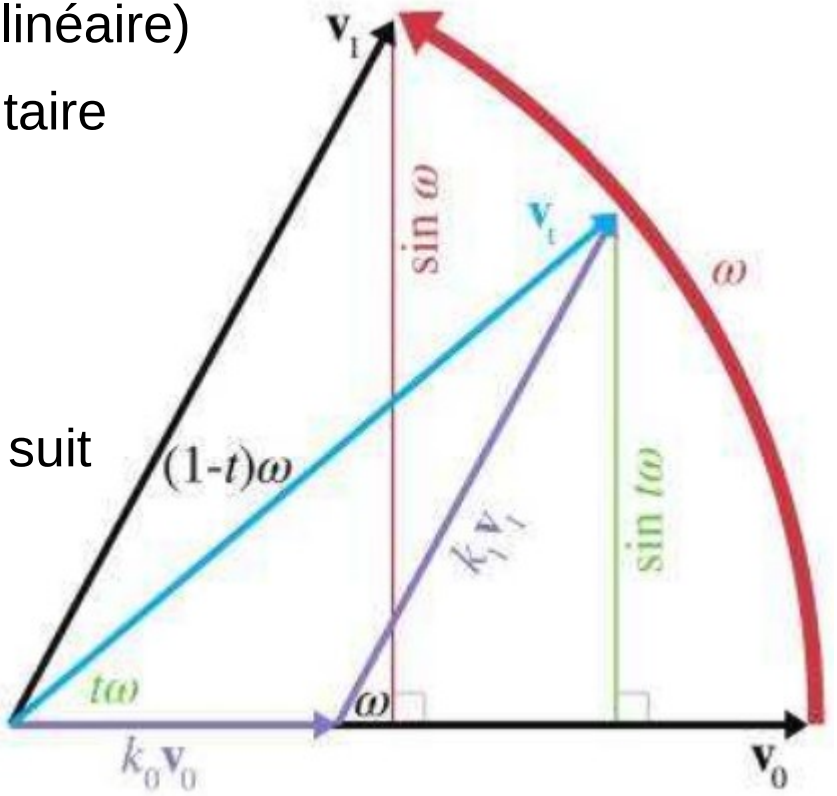
$$\sin\omega = \frac{\sin t\omega}{k_1} \qquad k_1 = \frac{\sin t\omega}{\sin\omega}$$

      - Même technique pour k_0

$$k_0 = \frac{\sin(1-t)\omega}{\sin\omega}$$

      - On peut donc exprimer $\mathbf{v_t}$ comme suit

$$\mathbf{v}_t = k_0\mathbf{v}_0 + k_1\mathbf{v}_1 = \frac{\sin(1-t)\omega}{\sin\omega}\mathbf{v}_0 + \frac{\sin t\omega}{\sin\omega}\mathbf{v}_1$$



Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire sphérique (SLERP)
    - Formule alternative (plus efficace)
      - On peut étendre le principe à l'espace des quaternions, et reformuler la formule SLERP comme suit

$$\mathrm{slerp}(\mathbf{q}_0, \mathbf{q}_1, t) = \frac{\sin(1-t)\omega}{\sin\omega}\mathbf{q}_0 + \frac{\sin t\omega}{\sin\omega}\mathbf{q}_1$$

      - Pour calculer "l'angle" ω qui est cette fois entre deux quaternions, il suffit de se souvenir que leur produit scalaire est égal à cos ω
      - Deux conditions toutefois
        - Choisir les signes de $\mathbf{q}_0$ et $\mathbf{q}_1$ de sorte que leur produit scalaire ne soit pas négatif
          - Ainsi, l'arc de rotation entre les deux sera le plus court
        - Si $\mathbf{q}_0$ et $\mathbf{q}_1$ sont très rapprochés, l'angle sera très petit, tout comme sin ω, et dans ce cas il faudra utiliser une simple LERP

Rotations et orientation en 3D

- Quaternions
  - Interpolation linéaire normalisée (NLERP)
    - LERP adaptée aux rotations
    - La formule est la suivante : $\mathrm{nlerp}(\mathbf{q}_0,\mathbf{q}_1,t) = \dfrac{(1-t)\mathbf{q}_0 + t\mathbf{q}_1}{\|(1-t)\mathbf{q}_0 + t\mathbf{q}_1\|}$
    - Avantages
      - Commutatif, contrairement à SLERP
      - Moins coûteux que SLERP (n'utilise ni sinus, ni cosinus)
    - Inconvénients
      - Cette interpolation ne prend pas en compte le fait que les quaternions sont des points dans une hypersphère 4D
      - En conséquence, contrairement à une interpolation SLERP, elle ne s'effectue pas à une vélocité constante
      - Ce qui peut conduire à des animations qui auront l'air trop rapides au milieu de la rotation, mais trop lentes à la fin

Rotations et orientation en 3D

- Quaternions
  - Avantages des quaternions
    - Interpolation "lisse" ("smooth") et "torque minimal"
    - Rapidité de concaténation et d'inversion de déplacements angulaires
    - Rapidité de conversion avec des matrices
    - Seulement 4 nombres (contre 9 pour une matrice)
  - Inconvénients des quaternions
    - Plus grands de 33% que les angles d'Euler
    - Peuvent devenir invalides en cas d'accumulation de mauvaises données (mais on peut les normaliser)
    - Difficulté à travailler avec (beaucoup moins intuitifs que les angles d'Euler, par exemple)

Rotations et orientation en 3D

- Quaternions
  - L'usage du SLERP fait débat
    - Exemple : **"Understanding Slerp, Then Not Using It"**, article de Jonathan Blow (*Braid*, *Witness*) publié en 2004 sur son blog (source)
      - Pour lui, SLERP est trop complexe et trop coûteux
      - On peut essayer de comprendre la technique...
      - ... afin d'éviter de l'utiliser (pour s'en remettre à NLERP)
    - Point de vue de Jason Gregory dans son livre
      - Les développeurs de Naughty Dog (*Uncharted*) ont trouvé qu'une bonne implémentation de SLERP était aussi performante que NLERP (tests effectués sur PS3)
      - Avoir SLERP aux côtés d'autres algorithmes (LERP, NLERP...) est une bonne chose. Il faut juste savoir lequel, dans un contexte donné, est le plus efficace

Rotations et orientation en 3D

- Comparaison entre les différentes méthodes
  - Rotations de points entre deux espaces de coordonnées
    - Matrices : possible et optimisable
    - Angles d'Euler : impossible
    - Quaternions : très difficile en pratique
  - Concaténation de plusieurs rotations
    - Matrices : possible et optimisable, avec risque d'erreurs
    - Angles d'Euler : impossible
    - Quaternions : possible, rapide, avec risque d'erreurs
  - Inversion de rotations
    - Matrices : facile et rapide avec la transposée
    - Angles d'Euler : pas facile
    - Quaternions : facile et rapide avec le conjugué

Rotations et orientation en 3D

- Comparaison entre les différentes méthodes
  - Interpolation
    - Matrices : extrêmement problématique
    - Angles d'Euler : possible, mais blocage de Cardan
    - Quaternions : possible, "lisse" et efficace grâce au SLERP
  - Interprétation "humaine" immédiate
    - Matrices : difficile
    - Angles d'Euler : immédiat
    - Quaternions : très difficile
  - Stockage (dans la mémoire ou dans un fichier)
    - Matrices : 9 nombres
    - Angles d'Euler : 3 nombres
    - Quaternions : 4 nombres

Rotations et orientation en 3D

- Comparaison entre les différentes méthodes
  - Représentation unique pour une rotation donnée
    - Matrices : oui
    - Angles d'Euler : non, en raison de l'aliasing
    - Quaternions : exactement deux pour chaque déplacement angulaire, chacun est la négation de l'autre
  - Possibilité d'être invalide
    - Matrices : oui
    - Angles d'Euler : jamais
    - Quaternions : oui

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion angles d'Euler → matrice
    - On considère les rotations autour des angles h (heading), p (pitch) et b (bank) comme des matrices de rotation

$$\mathbf{B} = \mathbf{R}_z(b) = \begin{bmatrix} \cos b & \sin b & 0 \\ -\sin b & \cos b & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{P} = \mathbf{R}_x(p) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos p & \sin p \\ 0 & -\sin p & \cos p \end{bmatrix}$$

$$\mathbf{H} = \mathbf{R}_y(h) = \begin{bmatrix} \cos h & 0 & -\sin h \\ 0 & 1 & 0 \\ \sin h & 0 & \cos h \end{bmatrix}$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion angles d'Euler → matrice
    - On utilisera les raccourcis suivants

$$ch = \cos h, \qquad cp = \cos p, \qquad cb = \cos b,$$
$$sh = \sin h, \qquad sp = \sin p, \qquad sb = \sin b.$$

    - On effectue les rotations dans le sens inverse

$$\mathbf{BPH} = \begin{bmatrix} ch\,cb + sh\,sp\,sb & sb\,cp & -sh\,cb + ch\,sp\,sb \\ -ch\,sb + sh\,sp\,cb & cb\,cp & sb\,sh + ch\,sp\,cb \\ sh\,cp & -sp & ch\,cp \end{bmatrix}$$

    - Pour l'opération inverse : $\mathbf{H^{-1}P^{-1}B^{-1}} = \mathbf{R_v}(-h)\mathbf{R_x}(-p)\mathbf{R_z}(-b) =$

$$\begin{bmatrix} ch\,cb + sh\,sp\,sb & -ch\,sb + sh\,sp\,cb & sh\,cp \\ sb\,cp & cb\,cp & -sp \\ -sh\,cb + ch\,sp\,sb & sb\,sh + ch\,sp\,cb & ch\,cp \end{bmatrix}$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion matrice → angles d'Euler (canoniques)

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$$

$$= \begin{bmatrix} \cos h \cos b + \sin h \sin p \sin b & \sin b \cos p & -\sin h \cos b + \cos h \sin p \sin b \\ -\cos h \sin b + \sin h \sin p \cos b & \cos b \cos p & \sin b \sin h + \cos h \sin p \cos b \\ \sin h \cos p & -\sin p & \cos h \cos p \end{bmatrix}$$

  - Calcul de p : $m_{32} = -\sin p$, d'où : $p = \arcsin(-m_{32})$
  - Calcul de h : $m_{31} = \sin h \cos p, \qquad m_{33} = \cos h \cos p,$

$$m_{31}/\cos p = \sin h, \qquad m_{33}/\cos p = \cos h.$$

$$h = \operatorname{atan2}(\sin h, \cos h) = \operatorname{atan2}(m_{31}/\cos p, m_{33}/\cos p)$$

  - Calcul de b : $m_{12} = \sin b \cos p \qquad m_{22} = \cos b \cos p$

$$b = \operatorname{atan2}(\sin b, \cos b) = \operatorname{atan2}(m_{12}/\cos p, m_{22}/\cos p)$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion quaternion → matrice
    - Quaternion de départ

$$\begin{bmatrix} w & \mathbf{v} \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) & \sin(\theta/2)\hat{\mathbf{n}} \end{bmatrix}$$

    - Matrice résultat

$$\begin{bmatrix} {n_x}^2(1-\cos\theta)+\cos\theta & n_x n_y(1-\cos\theta)+n_z\sin\theta & n_x n_z(1-\cos\theta)-n_y\sin\theta \\ n_x n_y(1-\cos\theta)-n_z\sin\theta & {n_y}^2(1-\cos\theta)+\cos\theta & n_y n_z(1-\cos\theta)+n_x\sin\theta \\ n_x n_z(1-\cos\theta)+n_y\sin\theta & n_y n_z(1-\cos\theta)-n_x\sin\theta & {n_z}^2(1-\cos\theta)+\cos\theta \end{bmatrix}$$

$$w = \cos(\theta/2) \qquad x = n_x\sin(\theta/2) \qquad y = n_y\sin(\theta/2) \qquad z = n_z\sin(\theta/2)$$

$$\begin{bmatrix} 1-2y^2-2z^2 & 2xy+2wz & 2xz-2wy \\ 2xy-2wz & 1-2x^2-2z^2 & 2yz+2wx \\ 2xz+2wy & 2yz-2wx & 1-2x^2-2y^2 \end{bmatrix}$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion matrice → quaternion
    - Matrice de départ **M** = $\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$
    - Quaternion résultat :

$$w=\frac{\sqrt{m_{11}+m_{22}+m_{33}+1}}{2} \quad \Longrightarrow \quad x=\frac{m_{23}-m_{32}}{4w} \quad y=\frac{m_{31}-m_{13}}{4w} \quad z=\frac{m_{12}-m_{21}}{4w}$$

$$x=\frac{\sqrt{m_{11}-m_{22}-m_{33}+1}}{2} \quad \Longrightarrow \quad w=\frac{m_{23}-m_{32}}{4x} \quad y=\frac{m_{12}+m_{21}}{4x} \quad z=\frac{m_{31}+m_{13}}{4x}$$

$$y=\frac{\sqrt{-m_{11}+m_{22}-m_{33}+1}}{2} \quad \Longrightarrow \quad w=\frac{m_{31}-m_{13}}{4y} \quad x=\frac{m_{12}+m_{21}}{4y} \quad z=\frac{m_{23}+m_{32}}{4y}$$

$$z=\frac{\sqrt{-m_{11}-m_{22}+m_{33}+1}}{2} \quad \Longrightarrow \quad w=\frac{m_{12}-m_{21}}{4z} \quad x=\frac{m_{31}+m_{13}}{4z} \quad y=\frac{m_{23}+m_{32}}{4z}$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion angles d'Euler → quaternion
    - Soient **h**, **p** et **b** les quaternions qui effectuent les rotations autour des axes respectifs *y*, *x* et *z*
    - On en déduit le quaternion résultat :

$$\mathbf{q}(h,p,b) = \mathbf{hpb} = \begin{bmatrix} \cos(h/2) \\ \begin{pmatrix} 0 \\ \sin(h/2) \\ 0 \end{pmatrix} \end{bmatrix} \begin{bmatrix} \cos(p/2) \\ \begin{pmatrix} \sin(p/2) \\ 0 \\ 0 \end{pmatrix} \end{bmatrix} \begin{bmatrix} \cos(b/2) \\ \begin{pmatrix} 0 \\ 0 \\ \sin(b/2) \end{pmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} \cos(h/2)\cos(p/2) \\ \begin{pmatrix} \cos(h/2)\sin(p/2) \\ \sin(h/2)\cos(p/2) \\ -\sin(h/2)\sin(p/2) \end{pmatrix} \end{bmatrix} \begin{bmatrix} \cos(b/2) \\ \begin{pmatrix} 0 \\ 0 \\ \sin(b/2) \end{pmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} \cos(h/2)\cos(p/2)\cos(b/2) + \sin(h/2)\sin(p/2)\sin(b/2) \\ \begin{pmatrix} \cos(h/2)\sin(p/2)\cos(b/2) + \sin(h/2)\cos(p/2)\sin(b/2) \\ \sin(h/2)\cos(p/2)\cos(b/2) - \cos(h/2)\sin(p/2)\sin(b/2) \\ \cos(h/2)\cos(p/2)\sin(b/2) - \sin(h/2)\sin(p/2)\cos(b/2) \end{pmatrix} \end{bmatrix}$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion quaternion → angles d'Euler
    - On peut d'abord convertir du quaternion à une matrice, puis de la matrice aux angles d'Euler
    - On aura besoin en particulier des cellules suivantes :

$$m_{11} = 1 - 2y^2 - 2z^2, \quad m_{12} = 2xy + 2wz, \quad m_{13} = 2xz - 2wy,$$
$$m_{22} = 1 - 2x^2 - 2z^2,$$
$$m_{31} = 2xz + 2wy, \quad m_{32} = 2yz - 2wx, \quad m_{33} = 1 - 2x^2 - 2y^2$$

Rotations et orientation en 3D

- Conversions entre les représentations
  - Conversion quaternion → angles d'Euler

$$
\begin{aligned}
p &= \arcsin(-m_{32}) \\
&= \arcsin\left(-2(yz - wx)\right)
\end{aligned}
$$

$$
h = \begin{cases}
\begin{aligned}
&\operatorname{atan2}(m_{31}, m_{33}) \\
&\quad = \operatorname{atan2}(2xz + 2wy, 1 - 2x^2 - 2y^2) \\
&\quad = \operatorname{atan2}(xz + wy, 1/2 - x^2 - y^2)
\end{aligned} & \text{si cos } p \neq 0 \\
\begin{aligned}
&\operatorname{atan2}(-m_{13}, m_{11}) \\
&\quad = \operatorname{atan2}(-2xz + 2wy, 1 - 2y^2 - 2z^2) \\
&\quad = \operatorname{atan2}(-xz + wy, 1/2 - y^2 - z^2)
\end{aligned} & \text{sinon}
\end{cases}
$$

$$
b = \begin{cases}
\begin{aligned}
&\operatorname{atan2}(m_{12}, m_{22}) \\
&\quad = \operatorname{atan2}(2xy + 2wz, 1 - 2x^2 - 2z^2) \\
&\quad = \operatorname{atan2}(xy + wz, 1/2 - x^2 - z^2)
\end{aligned} & \text{si cos } p \neq 0 \\
0 & \text{sinon}
\end{cases}
$$

Fin de la deuxième partie