Orientations et interpolation

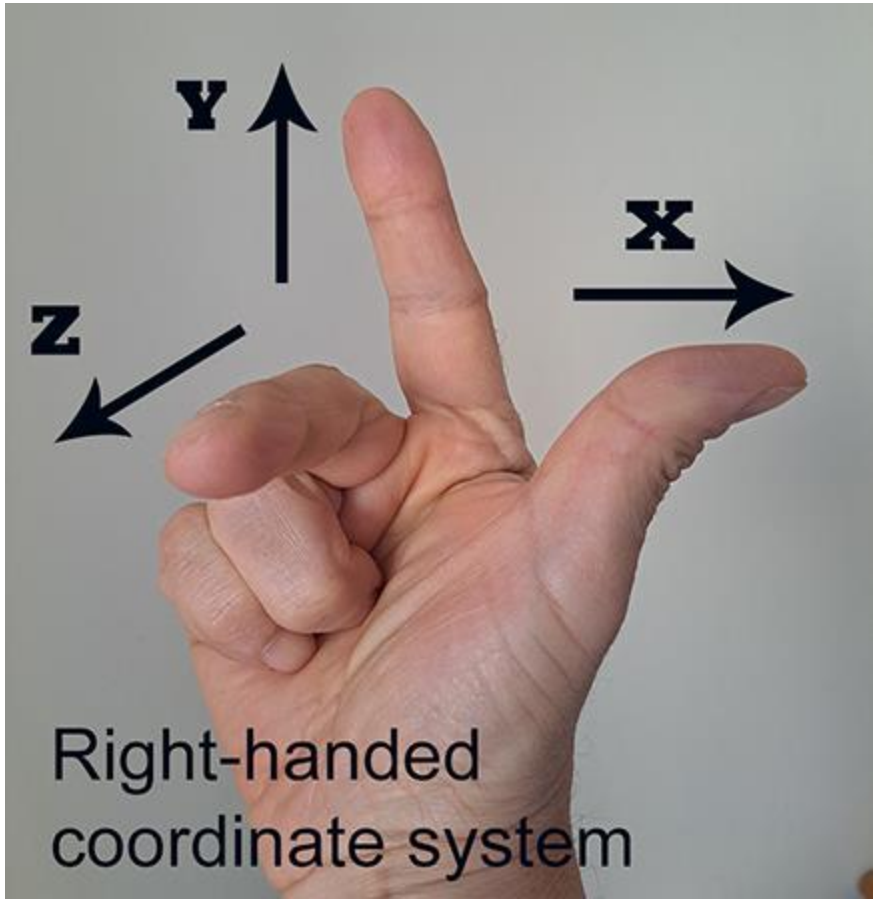
IMN504

UNIVERSITÉ DE SHERBROOKE

Transformations de l’espace

Repère 3D

- Matrices en coordonnées homogènes
  - Rotation
  - Translation
  - Changement d’échelle
  - Système main droite pour ce cours

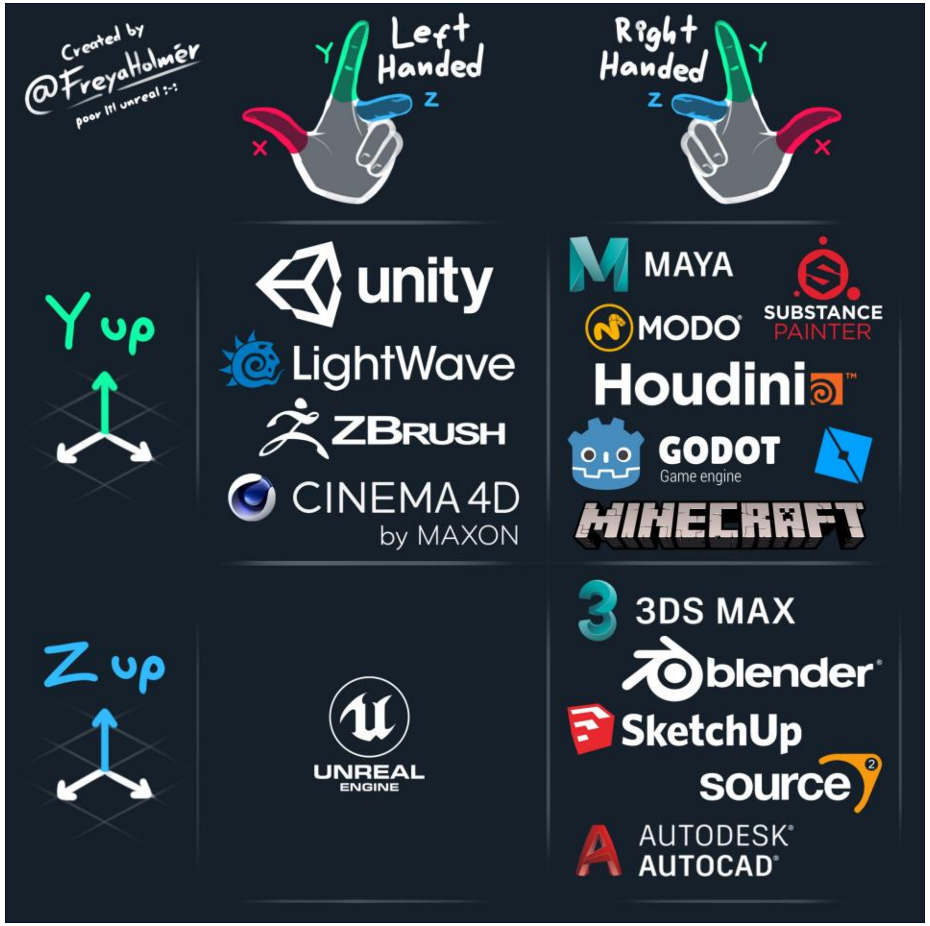


Transformations de l’espace

Repère 3D

- Des conventions propres a chaque logiciel
- Concepts identiques

- Attention aux conversions de format !



Rappel : coordonnées homogènes

Dans ce domaine projectif

- Quand $w = 1$, on est dans l'espace affine
- Quand $w = 0$, on est à l'infini
- Ce sera très utile pour les projections

Deux points sont dit égaux

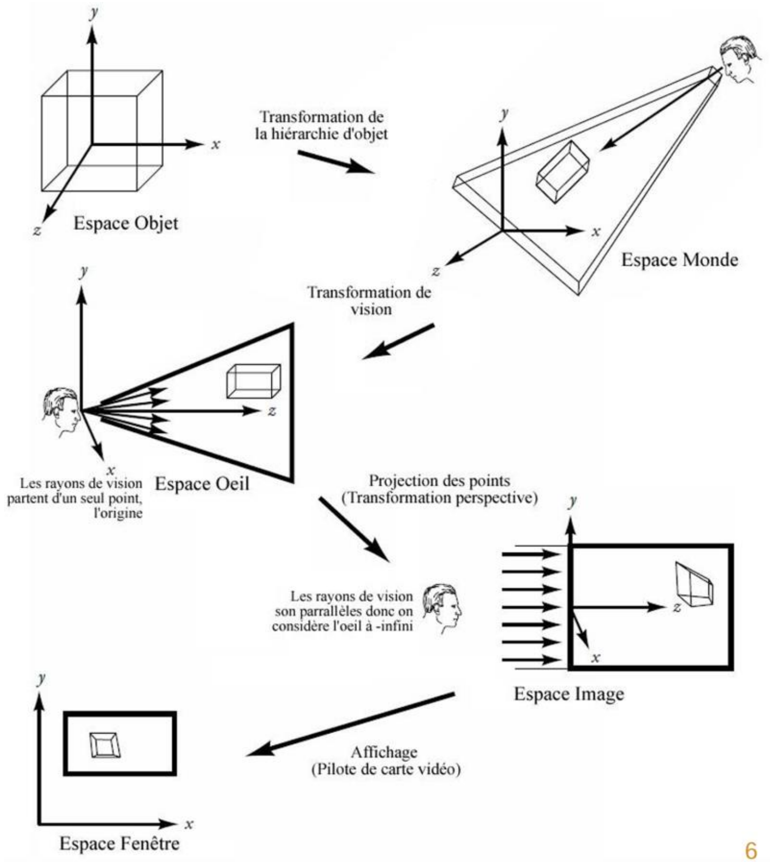
- SSI leurs coordonnées divisées par w sont égales

$$A = B \Leftrightarrow \begin{bmatrix} \frac{x_A}{w_A} \\ \frac{y_A}{w_A} \end{bmatrix} = \begin{bmatrix} \frac{x_B}{w_B} \\ \frac{y_B}{w_B} \end{bmatrix}$$

Transformations de l'espace

Espace et repère

- Espace objet
- Espace Monde
- Espace Œil(caméra)
- Espace Image
- Espace fenêtre



Parent, R. : "Computer Animation : Algorithms and techniques", 1st edition, Morgan Kaufmann, 2002, p.37

En pratique

Vertex Shader

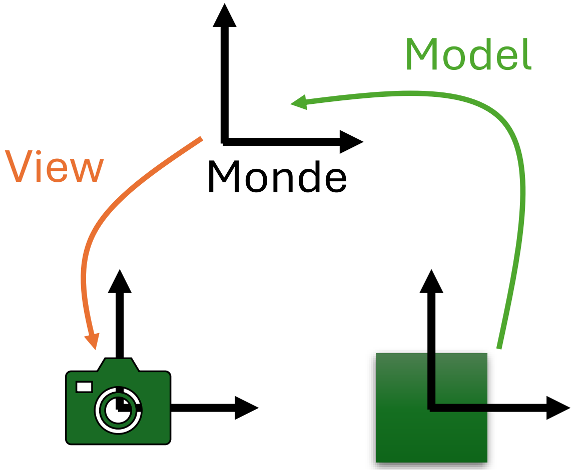
```
void main()
{
    gl_Position = Proj * View * Model * vec4(Position,1.0);
}
```

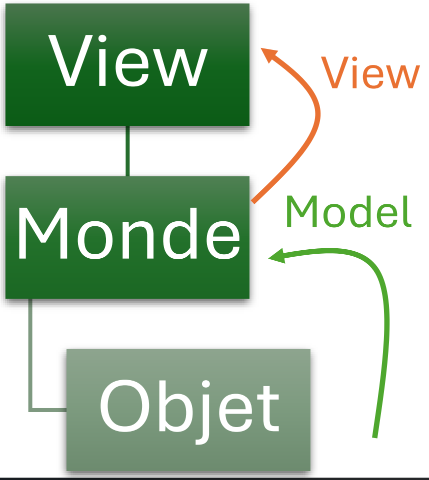
Espace Objet

Espace Monde

Espace Caméra

Espace Image





Transformations affines

Rigide

- Préserve forme et taille
- Translation, rotation

Similarité

- Préserve la forme
- Translation, rotation
- Changement d’échelle uniforme

Quelconque

- Ne préserve pas les caractéristiques
- Cisaillement, changement non-uniforme

Transformations affines (en 2d)

Translation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_T P$
Rotation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_R P$
Changement d'échelle	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_x & 0 & 0 \\ 0 & \Delta_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_S P$
Cisaillement	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_C P$

Matrice de transformation

Anatomie d’une matrice de transformation affine 3D

$$M = \begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation, cisaillement, échelle

Translation

Gestion du temps

Gestion du temps

Cas d'usage : une scène dynamique

- Dans une scène classique, chaque objet peut être en mouvement à tout instant
  - On applique des transformations
- Une transformation est appliquée des dizaines de fois par seconde
- Impact :
  - Performance
  - Précision
  - Contrôle

Question pratique

- Comment gérer ces transformations ?
- Que stocker ? Que calculer ?
- Quelles performances ?

Quelles sont les limites de nos outils ?

La précision

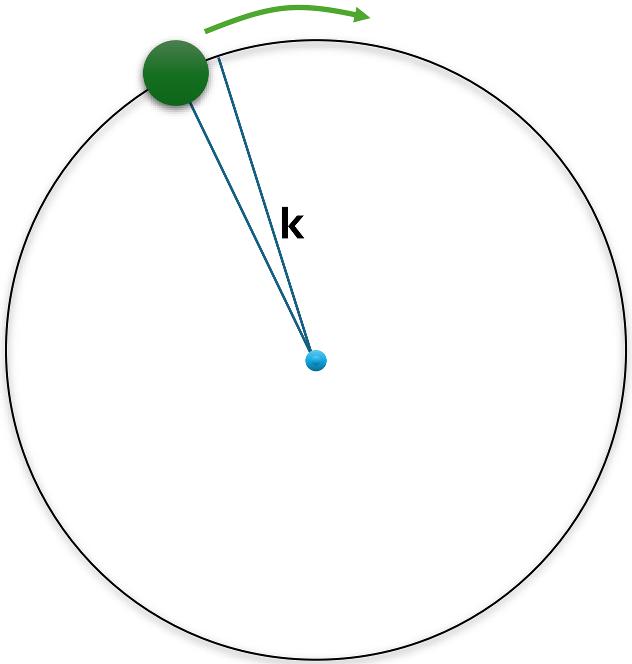
Erreur d’arrondi accumulé

- Une transformation est appliquée des dizaines de fois par seconde
- Mais ...nos informations sont discrètes
- Cela va conduire à des erreurs d’arrondi qui vont s’accumuler

Exemple

Rotation d'une sphère

- Soir **S** une sphère autour d'un point
- Rotation R_k de **k** degrés par image / pas de temps



Exemple

Méthode 1

- Appliquer R_k à chaque sommet de **S** de la sphère image après image

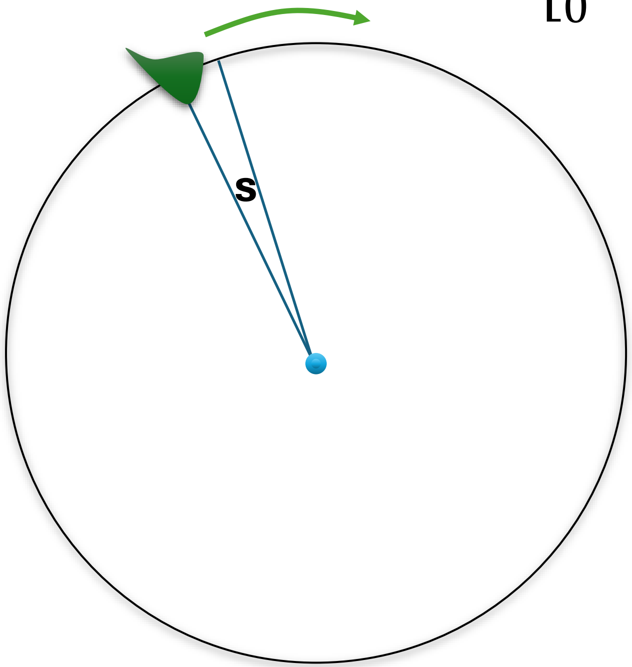
Problème

- Déformation progressive de la sphère
- Les points ne sont plus coplanaires

Erreur d'arrondi

- L'erreur s'accumule dans les sommets de la sphère

$$M = \begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Exemple

Méthode 2

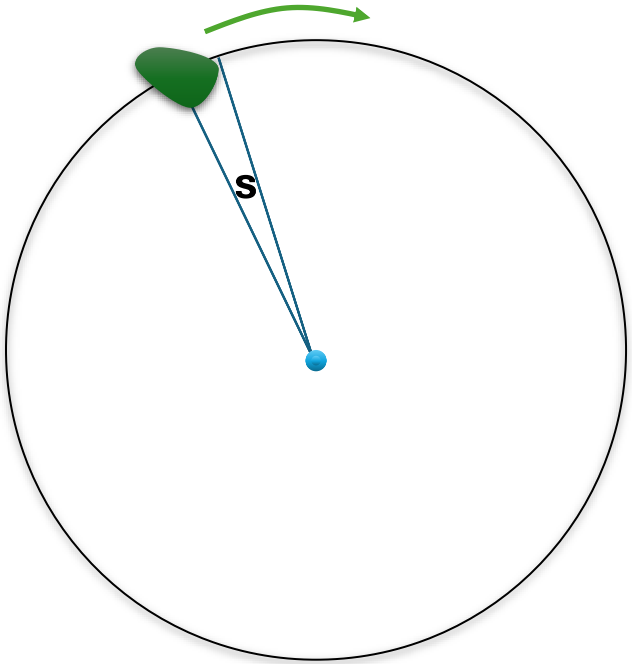
- Créer une matrice de transformation M
- Appliquer la rotation R_kà la matrice M
- Multiplier chaque sommet original de **S** par M

Problème

- Cisaillement progressif de la sphère
- Erreur au niveau de la rotation ou l'échelle de la sphère

Erreur d'arrondi

- L'erreur s'accumule dans la partie supérieure gauche de M (changement d'échelle,orientation)



Exemple

Méthode 3

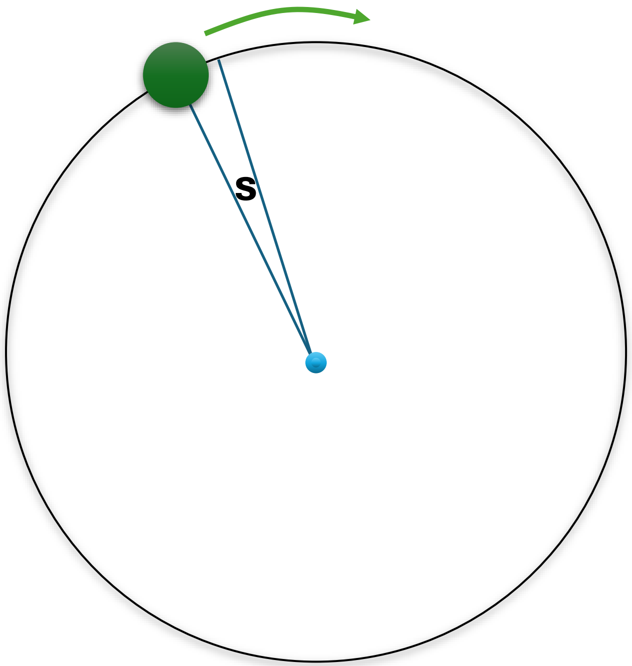
- Incrémenter une variable **p** de **k** à chaque pas de temps
- Créer la matrice M_P de la rotation R_p
- Multiplier chaque sommet original de **S** par M_P

Problème

- Légère perte de précision après longtemps

Erreur d'arrondi

- L'erreur s'accumule dans la variable **p**
- L'objet ne se déforme pas



Transformations au cours du temps

De manière générale

- Affecter le moins d'éléments possibles
- Ne pas affecter les éléments visibles
- Éviter les incrémentations successives d'un grand nombre de valeurs
- Possible d'orthonormaliser les matrices de transformation

Animation et interpolation

Interpolation

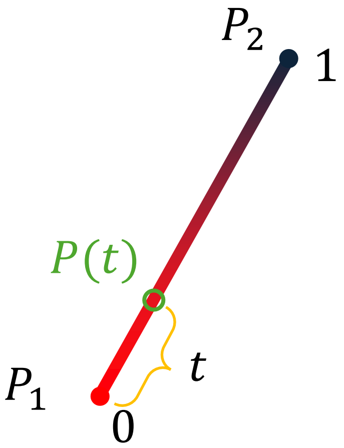
Besoin d’interpolations

- Souvent en animation, on raisonne par keyframe (cadre clé)
- On positionne un objet à un ensemble de points spécifiques
- Om effectue ensuite une interpolation entre les keyframe

Interpolation

- Interpoler correctement n’est pas toujours facile
- Dépend des outils
- Et de ce qu’on interpole

Interpolation linéeaire

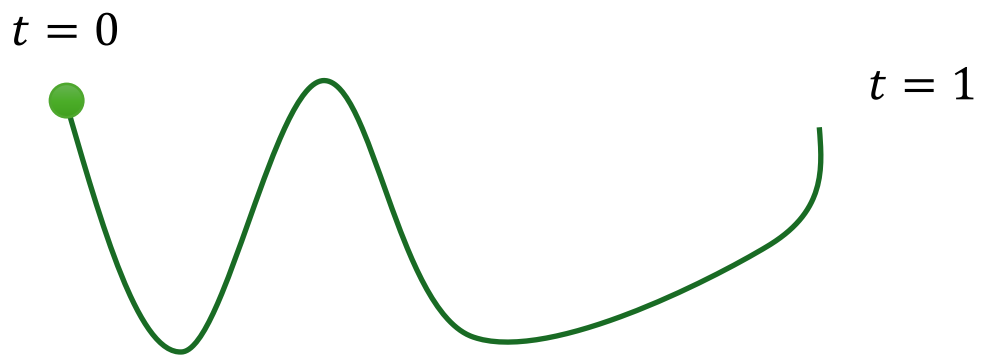


Interpolation linéaire

- Interpolation entre deux positions selon

$\epsilon[0,1]$

Interpolation paramétrique



Courbe $C(t)$ caractérisée par un paramètre t

- Un polynôme de degré n

$$x(t) = C_x(t)$$
$$y(t) = C_y(t)$$

Souvent des trajectoires plus naturelles

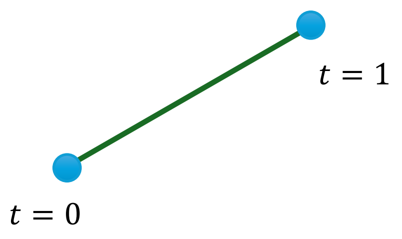
- Paramètres de C
- Dépend du degré de polynôme

Courbes paramétriques

Différents degrés du polynôme de définition

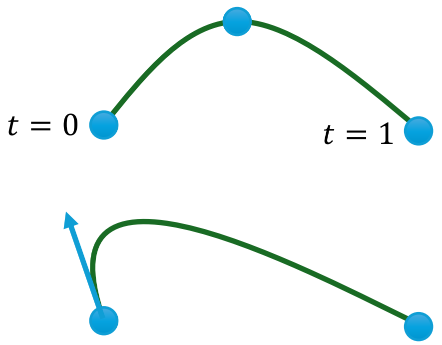
Linéaire ($n = 1$)

- $C(t) = at + b$
- Deux contraintes



Quadratique ($n = 2$)

- $C(t) = at^2 + bt + c$
- Trois contraintes
  - Trois points
  - Deux points et une tangente



Courbes paramétriques

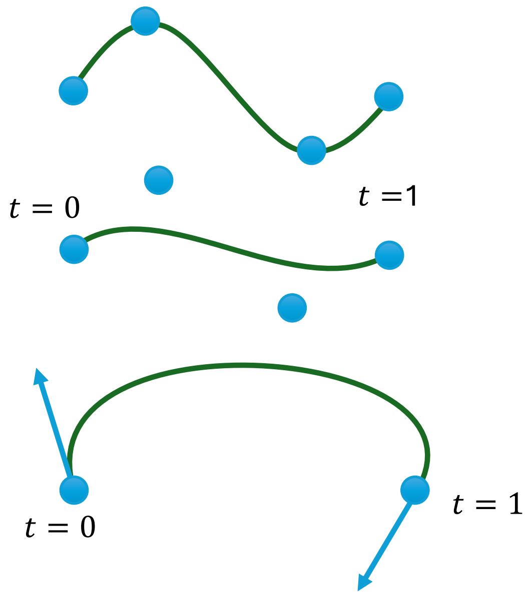
Différents degrés du polynôme de définition

Cubique($n = 3$)

- $C(t) = at^3 + bt^2 + ct + d$
- Quatre contraintes
  - Quatre points
  - Deux points et deux tangentes

Plus haut degré ($n > 3$)

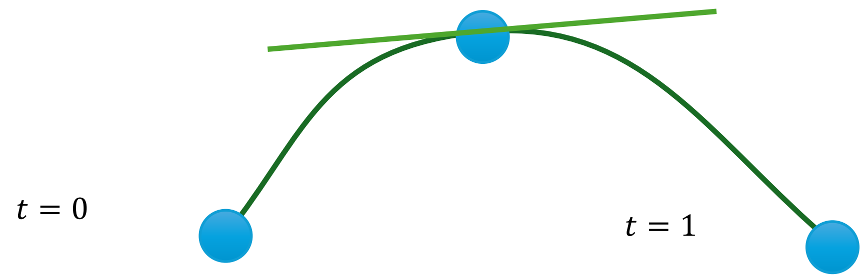
- $C(t) = \sum p_i t^i$
- $n + 1$ contraintes
- Difficile à contrôler
- Oscillations



Position et tangente

Tangente d'une courbe

- Dérivée de $C(t) = at^3 + bt^2 + ct + d$
- $C'(t) = \frac{\partial C(t)}{\partial t} = 3at^2 + 2bt + c$

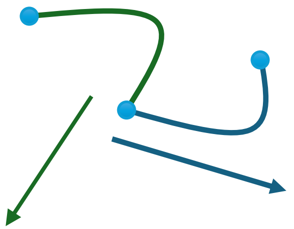


Continuité des courbes

Il existe plusieurs niveaux de continuité pour les courbes

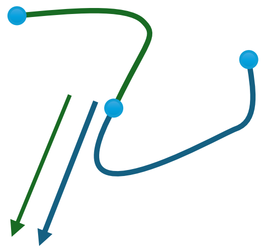
Continuité C^0

- Continuité positionnelle



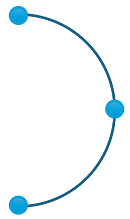
Continuité C^1

- Continuité positionnelle
- Continuité tangentielle (1ere dérivée)



Continuité C^2

- Continuité positionnelle
- Continuité tangentielle (1ere dérivée)
- Continuité de courbure (2eme dérivée)



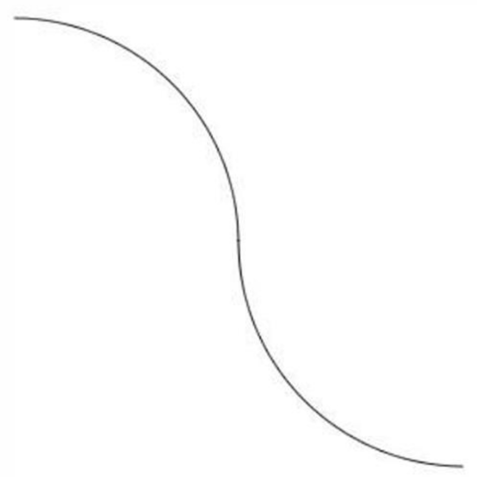
Continuité C^n

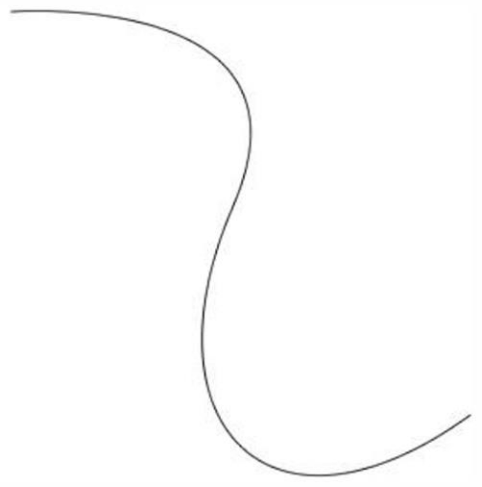
- Les $n^{èmes}$ dérivées de C sont égales

Raccordement des courbes

Difficile d'observer la courbure à l'œil nu

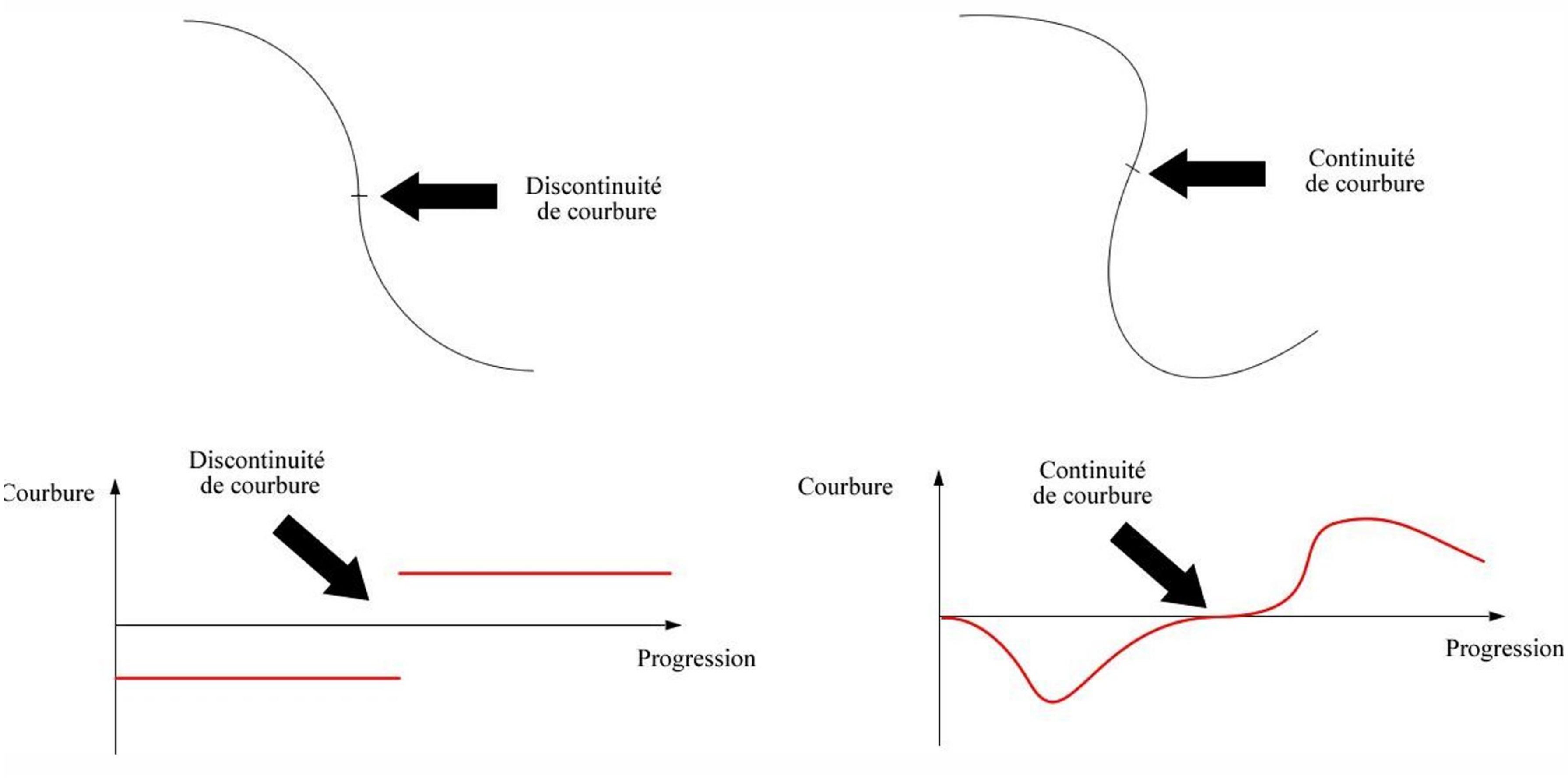
Ces courbes sont elles C^2 ?





Raccordement des courbes

Difficile d’observer la courbure à l’œil nu



Courbes de hermite

Courbes de Hermite

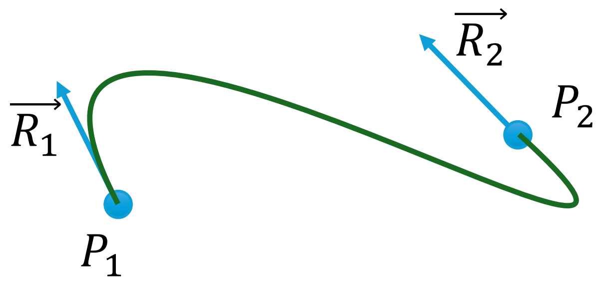
- D'après le mathématicien français **Charles Hermite**
- Courbe **interpolante**

Les contraintes

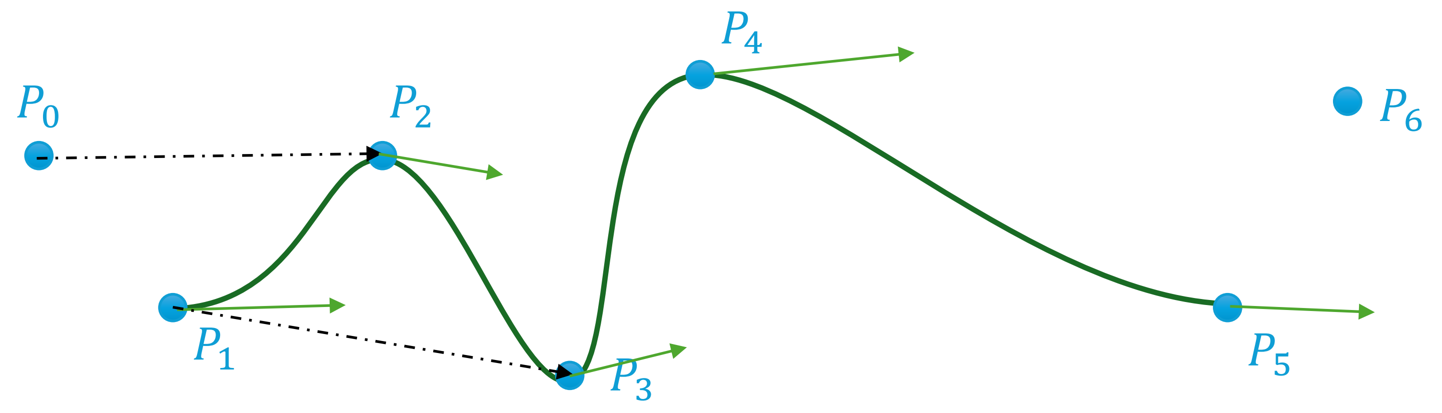
- Deux points aux extrémités
  - P_1 et P_2
- Deux tangentes aux extrémités
  - $\overrightarrow{R_1}$ et $\overrightarrow{R_2}$

Fonctions de mélange

- $C(t)$ est une moyenne pondérée entre les contraintes
- $C(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix}$
- $C(t) = \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix}$



Courbes de Catmull-ROM



Courbes de Catmull-Rom

- Tangentes définies par les points de contrôle

Forme matricielle

- $C(t) =$

$$[t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix} -s & 2-s & s-2 & s \\ 2s & s-3 & 3-2s & -s \\ -s & 0 & s & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_{i-2} \\ P_{i-1} \\ P_i \\ P_{i+1} \end{bmatrix}$$

Courbes de Bezier

Courbes de Bezier

- D'après Pierre Bezier
- Courbe **approximante**

Les contraintes

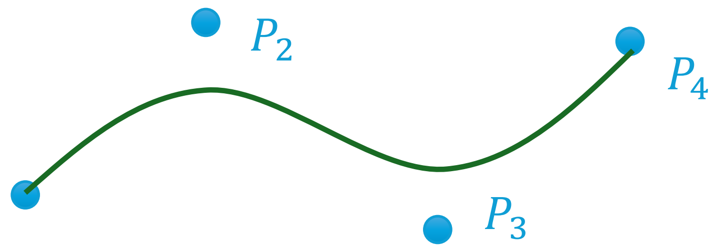
- Deux points aux extrémités
  - P_1 et P_2
- Tangentes définies par les points de contrôle

Forme matricielle

(en degré 3)

- $C(t) =$

$$[t^3 \quad t^2 \quad t \quad 1] \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$



Interpolation quadratique ou cubique

Orientations et rotations

Représenter L'orientation

Idéalement

- Définition intuitive
- Couvrir toutes les configurations
- Complexité de calcul réduite
- Interpolation efficace
- Une formulation générique

Différents outils

- Capacité de représentation
- Complexité
- Interpolation
- Composition

Représenter l'orientation

Matrices

- $R = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix}$
- $R^T R = I$
- $\det(R) = I$

Angles d'Euler

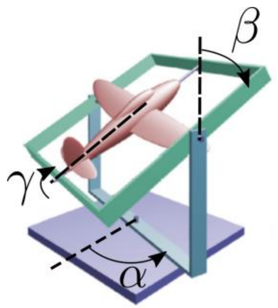
- 3 angles (α, β, γ)
- Composition de rotations autour des axes

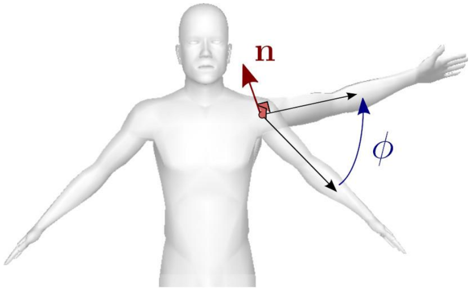
Angle d'axe

- $(\boldsymbol{n}, \theta)$
- Rotation autour d'un axe
- Formule de Rodrigues

Quaternions

- $q = (s, x, y, z)$
  $= (\cos\left(\frac{\theta}{2}\right), \boldsymbol{n} \sin\left(\frac{\theta}{2}\right))$





3 degrés de liberté

Matrice de rotation

Translation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_T P$
Rotation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_R P$
Changement d'échelle	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_x & 0 & 0 \\ 0 & \Delta_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_S P$
Cisaillement	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	$P' = M_C P$

Matrices de rotation

Interpolation de rotations

- Interpoler des matrices n'est pas une bonne idée
- Exemple : rotations autour de Y
- $R1(90°) \quad = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, R2(-90°) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$
- $0.5 * R1 + 0.5 * R2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
- C'est une transformation non inversible, non rigide
  - Echec de l'interpolation

Représenter l'orientation

Matrices

- $R = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix}$
- $R^T R = I$
- $\det(R) = I$

Angles d'Euler

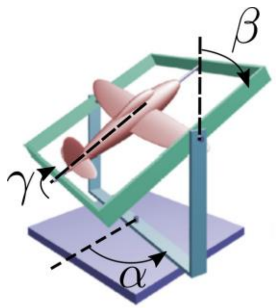
- 3 angles (α, β, γ)
- Composition de rotations autour des axes

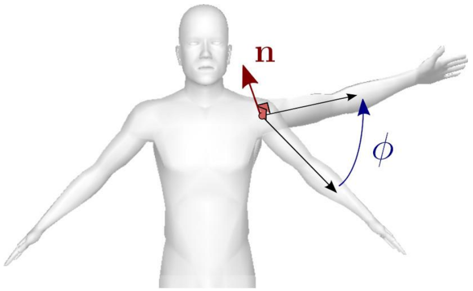
Angle d'axe

- $(\boldsymbol{n}, \theta)$
- Rotation autour d'un axe
- Formule de Rodrigues

Quaternions

- $q = (s, x, y, z)$
  $= (\cos\left(\frac{\theta}{2}\right), \boldsymbol{n} \sin\left(\frac{\theta}{2}\right))$



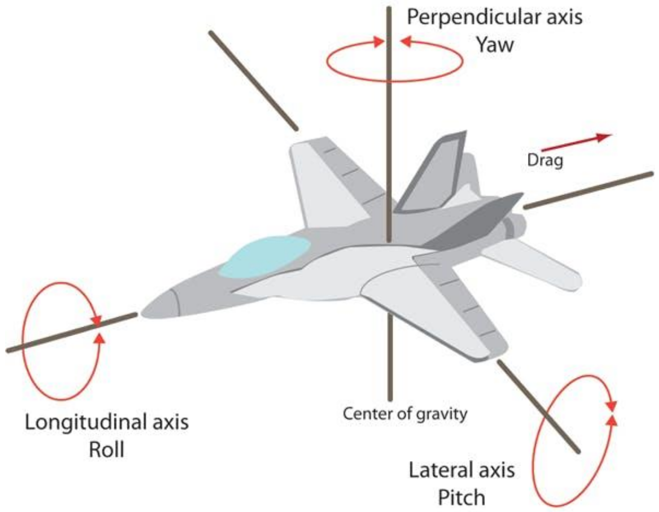


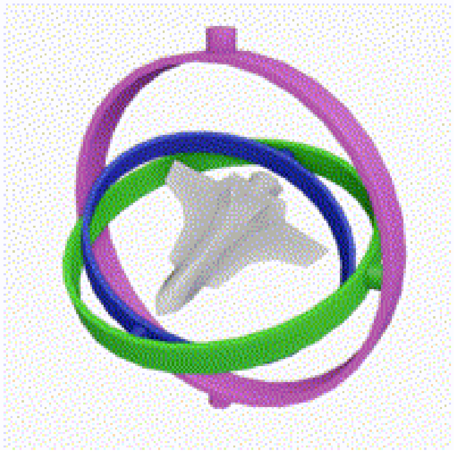
3 degrés de liberté

Rotation par les angles d'Euler

Angles d'Euler

- Rotations définies par trois valeurs relatives au repère local
- Lacet, roulis, tangage
- Problème de **Gimbal lock**
  - Les rotations dépendent les unes des autres
  - On peut perdre un degré de liberté
    - On ne peut plus revenir en arrière
  - Plusieurs solutions possibles pour aller d'une orientation à une autre





Représenter l’orientation

Matrices

- $R = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix}$
- $R^T R = I$
- $\det(R) = I$

Angles d’Euler

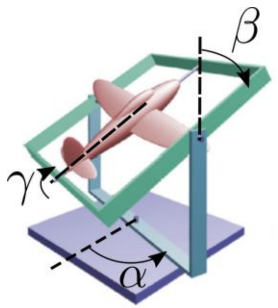
- 3 angles (α, β, γ)
- Composition de rotations autour des axes

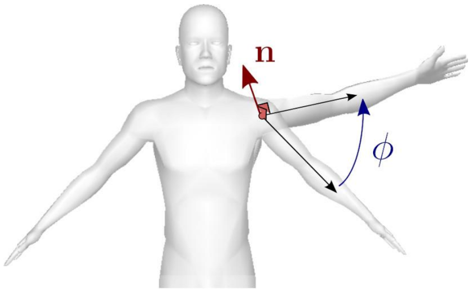
Angle d’axe

- $(\boldsymbol{n}, \theta)$
- Rotation autour d’un axe
- Formule de Rodrigues

Quaternions

- $q = (s, x, y, z)$
  $= (\cos\left(\frac{\theta}{2}\right), \boldsymbol{n} \sin\left(\frac{\theta}{2}\right))$





3 degrés de liberté

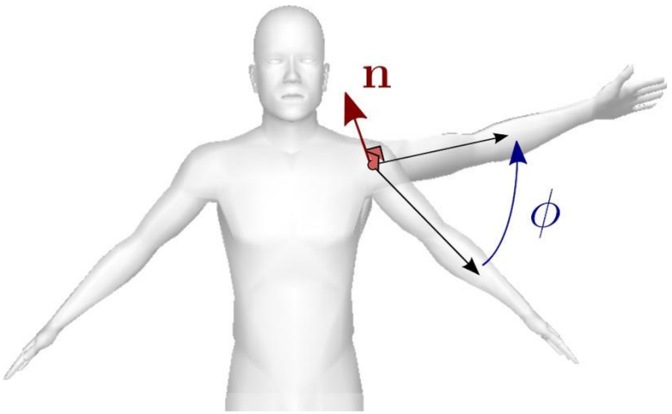
Rotation par angles et axe

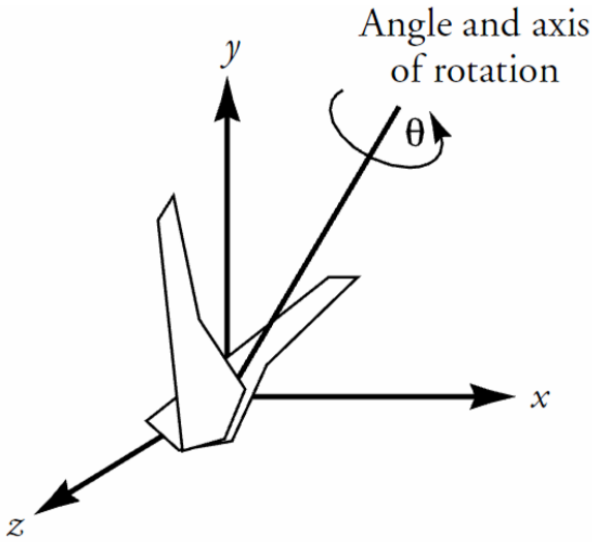
Angle d’axes

- Rotations définies par un axe de rotation et une valeur d’angle
- Simple et concis

Formule de Rodrigues

- Calculer la transformation
  - Revenir à une matrice de rotation





Méthode de Rodrigues

Matrices antisymétriques

$$A = -A^T$$

$$A = \begin{bmatrix} 0 & a & b \\ -a & 0 & c \\ -b & -c & 0 \end{bmatrix}$$

Méthode de Rodrigues

Rappel du produit vectoriel

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

Soit la matrice antisymétrique

$$\hat{a} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

Alors

$$\hat{a} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} -a_3 b_2 + a_2 b_3 \\ a_3 b_1 - a_1 b_3 \\ -a_2 b_1 + a_1 b_2 \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Méthode de Rodrigues

La matrice de rotation associée à un axe de rotation s et un angle de rotation θ

$$R = I + \sin(\theta)\hat{s} + (1 - \cos(\theta))\hat{s}^2$$

Détail de la preuve : https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula

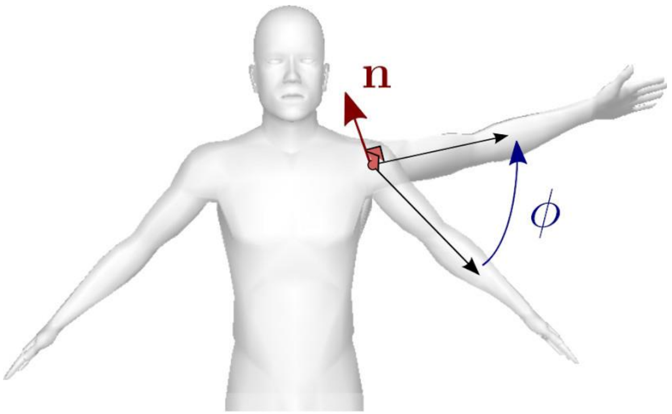
Rotation par angles et axe

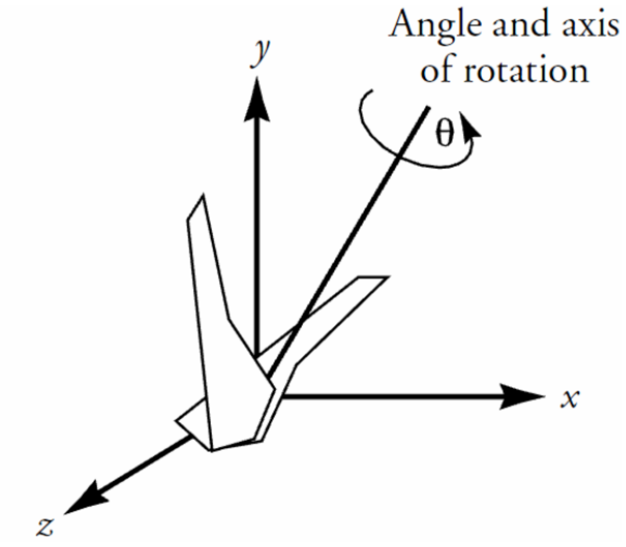
Angle d'axes

- Rotations définies par un axe de rotation et une valeur d'angle
- Simple et concis
- Formule de Rodrigues

Interpolation ?

- Mais pas d'interpolation directe
- Composition de rotation difficile



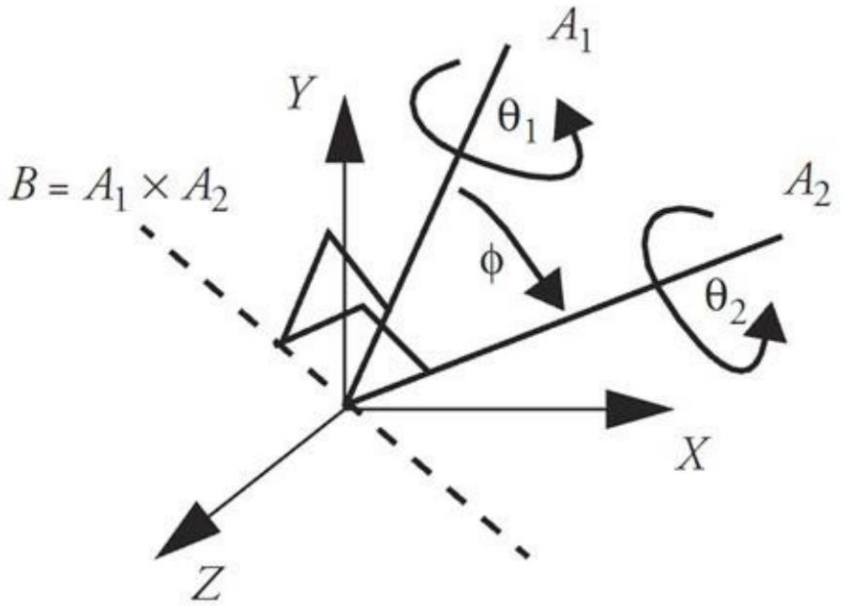


Rotation par angles et axe

Interpolation

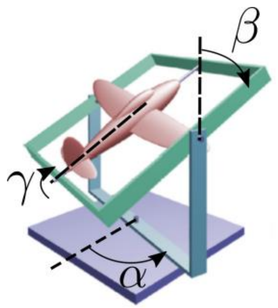
- Interpoler les vecteurs d'axe, puis les angles
- Ex : $(A1, \theta 1) \rightarrow (A2, \theta 2)$
- Trouver un axe de rotation B entre A1 et A2 : $B = A1 \times A2$
- A chaque pas de temps k
- Trouver l'axe $Ak = Rot(B, k\phi, A1)$
  - avec $\phi = \cos^{-1}(\frac{A1 \cdot A2}{|A1||A2|})$
- Interpolation l'orientation entre $\theta 1$ et $\theta 2$

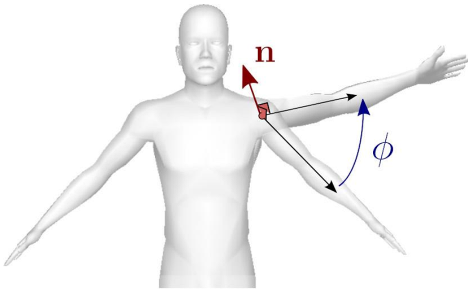
- Peu pratique à implémenter
- Interpolation entre plusieurs orientations ?



Représenter l'orientation

Matrices	Angles d'Euler	Angle d'axe	Quaternions
• $R = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix}$ • $R^T R = I$ • $\det(R) = I$	• 3 angles (α, β, γ) • Composition de rotations autour des axes	• $(\boldsymbol{n}, \theta)$ • Rotation autour d'un axe • Formule de Rodrigues	• $q = (s, x, y, z)$ $= (\cos\left(\frac{\theta}{2}\right), \boldsymbol{n} \sin\left(\frac{\theta}{2}\right))$





3 degrés de liberté

Rotations : Les quaternions

Généralisation des nombres complexes

Définition sur une hypersphère (dimension 4)

$$q = s + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} = (\boldsymbol{s}, \boldsymbol{v})$$

- avec s partie réelle et (i, j, k) partie imaginaire

Propriétés algébriques

- $i^2 = j^2 = k^2 = -1$
- $ij = -ji = k$
- $jk = -kj = \mathrm{i}$
- $ki = -ik = j$
- $ijk = -1$

Opérations basiques

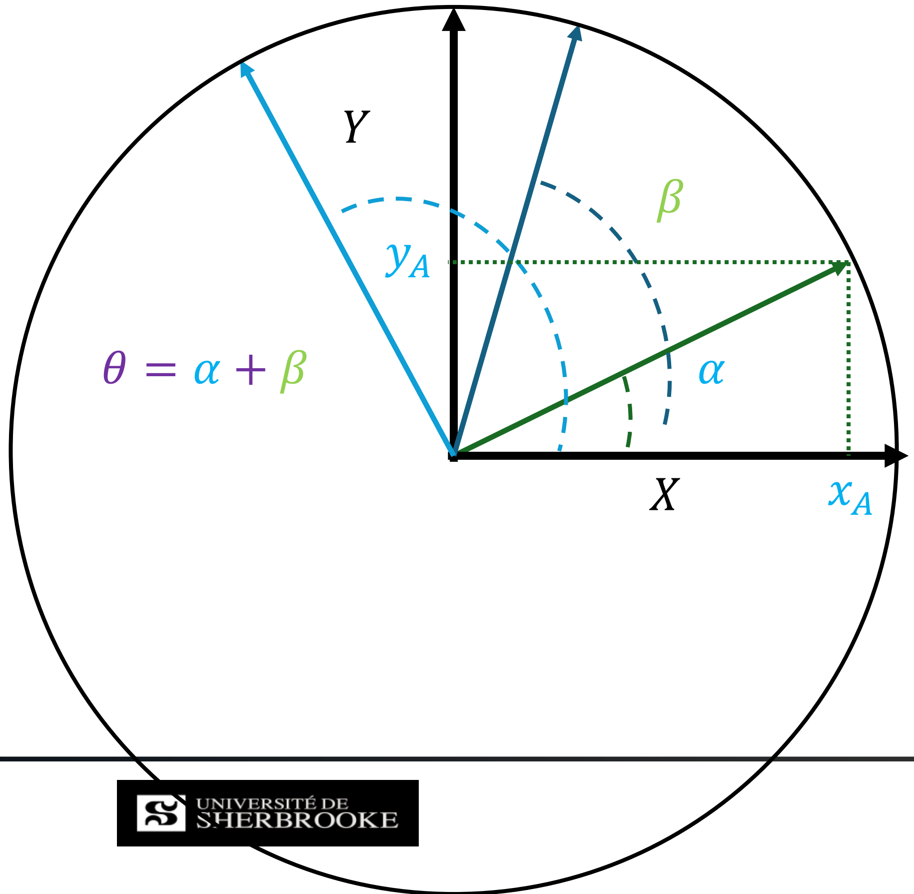
- Conjugué : $q^* = (s, -x, -y, -z)$
- Norme: $\|q\| = \sqrt{qq^*} = \sqrt{s^2 + x^2 + y^2 + z^2}$
- Quaternion unité : $\|q\| = 1$

Commencons en 2D

$\vec{A} = x_A\vec{X} + y_A\vec{Y} = \cos\alpha\,\vec{X} + \sin\alpha\,\vec{Y} = (x_A, y_A)$

$\vec{B} = x_B\vec{X} + y_B\vec{Y} = \cos\beta\,\vec{X} + \sin\beta\,\vec{Y} = (x_B, y_B)$

$\vec{G} = x_G\vec{X} + y_G\vec{Y} = (\cos(\alpha+\beta)\,\vec{X} + \sin(\alpha+\beta)\,\vec{Y}) = ?$



Nombres complexes (2D)

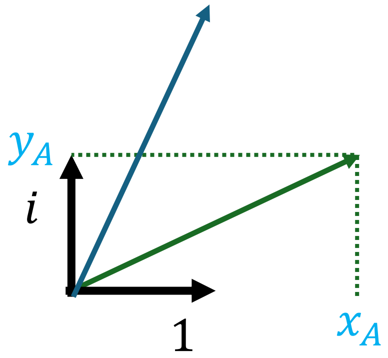
$A = x_A 1 + y_A \boldsymbol{i} = x_A + y_A \boldsymbol{i} = (x_A, y_A)$

$B = x_B 1 + y_B \boldsymbol{i} = x_B + y_B \boldsymbol{i} = (x_B, y_B)$

Nouvelle algèbre

$\boxed{i^2 = -1}$

$$
\begin{aligned}
AB &= (x_A + y_A \boldsymbol{i})\,(x_B + y_B \boldsymbol{i}) \\
&= x_A\, x_B + x_A\, y_B \boldsymbol{i} + y_A x_B \boldsymbol{i} + y_A\, y_B\, \boldsymbol{ii} \\
&= x_A\, x_B - y_A\, y_B + (x_A\, y_B + y_A x_B)\, \boldsymbol{i} \\
&= (x_A\, x_B - y_A\, y_B\, , x_A\, y_B + y_A x_B)
\end{aligned}
$$



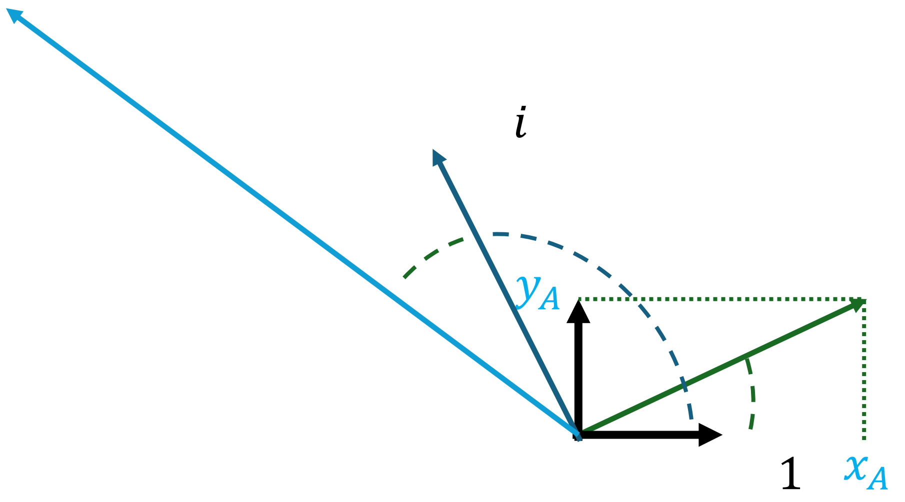
Rotation et Nombres complexes

$$\boxed{i^2 = -1}$$

$A = 2 + \boldsymbol{i}$

$B = -1 + 2\,\boldsymbol{i}$

$C = AB = (2 + \boldsymbol{i})\,(-1 + 2\boldsymbol{i})$
$\quad = -2 + 4\boldsymbol{i} - 1\boldsymbol{i} + 2\,\boldsymbol{i}\boldsymbol{i}$
$\quad = -4 + 3\boldsymbol{i}$



Rotation et Nombres complexes

Algèbre des complexes

- Multiplication de deux vecteurs
- Un nouveau vecteur
  - Addition des angles
  - Multiplication des normes
    - $|C| = |A||B|$
- Si un des deux vecteurs est unitaire
  - Cela applique une rotation

Et en 3D ?

- Opération plus difficile
- Plusieurs plans de rotation (un seul en 2D)

Rotations : Les quaternions

Passer en 3D

Ajouter **j** et **k**

4 composantes

$$q = s + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} = (\boldsymbol{s}, \boldsymbol{v})$$

- avec s partie réelle et (i, j, k) partie imaginaire

Propriétés algébriques

- $i^2 = j^2 = k^2 = -1$
- $ij = k$
- $jk = i$
- $ki = j$
- $ijk = -1$
- $ij = -k, kj = -i\ldots$

Opérations basiques

- Conjugué : $q^* = (s, -x, -y, -z)$
- Norme: $\|q\| = \sqrt{qq^*} = \sqrt{s^2 + x^2 + y^2 + z^2}$
- Quaternion unité : $\|q\| = 1$

Rotations : Les quaternions

Généralisation des nombres complexes

Définition sur une hypersphère (dimension 4)

$$q = s + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} = (\boldsymbol{s}, \boldsymbol{v})$$

- avec s partie réelle et (i, j, k) partie imaginaire

Propriétés algébriques

- $i^2 = j^2 = k^2 = -1$
- $ij = k$
- $jk = i$
- $ki = j$
- $ijk = -1$
- $ij = -k, kj = -i\ldots$

Opérations basiques

- Conjugué : $q^* = (s, -x, -y, -z)$
- Norme: $\|q\| = \sqrt{qq^*} = \sqrt{s^2 + x^2 + y^2 + z^2}$
- Quaternion unité : $\|q\| = 1$

Rotations : Les quaternions

Une sphère en 4 dimensions

- Visualisation difficile
- Projection stéréographique en 3D
- Ressource pour l'intuition https://eater.net/quaternions

Rotations : Les quaternions

Produit de quaternions

- $q_1q_2 = (s_1 + x_1\boldsymbol{i} + \ y_1\boldsymbol{j} + z_1\boldsymbol{k})(s_2 + x_2\boldsymbol{i} + y_2\boldsymbol{j} + z_2\boldsymbol{k})$
- $q_1q_2 = \begin{pmatrix} s_1s_2 - x_1x_2 - y_1y_2 - z_1z_2 \\ x_1s_2 + s_1x_2 + y_1z_2 - z_1y_2 \\ y_1s_2 + s_1y_2 + z_1x_2 - x_1z_2 \\ z_1s_2 + s_1z_2 + x_1y_2 - y_1x_2 \end{pmatrix}$
- $q_1q_2 = (s_1, v_1)(s_2, v_2) = (s_1s_2 - v_1 \cdot v_2, v_1s_2 + v_2s_1 + v_1 \times v_2)$

Rotations : Les quaternions

Relation avec la rotation

- Un quaternion unitaire $q = (s, v)$
- Un vecteur $u = (u_x, u_y, u_z)$ assimilé au quaternion pur $q_u = (0, u) = (0, u_x, u_y, u_z)$
- Alors
  - $q_{u'} = \mathcal{R}_q(u) = q q_u q^*$ est un quaternion pur $q_{u'} = (0, u'_x, u'_y, u'_z)$
  - Et u' est la rotation de u autour de l'axe $n = \frac{v}{\|v\|}$ d'angle $2 * \cos^{-1}(s)$
  - Le quaternion unitaire $q = (\cos\left(\frac{\theta}{2}\right), n \sin\left(\frac{\theta}{2}\right))$ est la rotation d'angle θ autour de n

Rotations : Les quaternions

Composition de rotations

- Soit 2 rotations $(\mathcal{R}_1, \mathcal{R}_2)$ avec leurs quaternions unitaires (q_1, q_2)
- Le produit $q_1 q_2$ est la composition $\mathcal{R}_1 \circ \mathcal{R}_2$
- $\mathcal{R}_{q_1 q_2}(u) = (q_1 q_2) u (q_1 q_2)^*$
- $\mathcal{R}_{q_1 q_2}(u) = (q_1 q_2) u (q_2^* q_1^*)$, car $(q_1 q_2)^* = (q_2^* q_1^*)$
- $\mathcal{R}_{q_1 q_2}(u) = q_1 (q_2 u q_2^*) q_1^* = q_1 \mathcal{R}_{q_2} q_1^* = \mathcal{R}_{q_1} \circ \mathcal{R}_{q_2}(u)$

Rotations : Les quaternions

Matrice de rotation

- Un quaternion unitaire $q = (s, x, y, z)$ représente la matrice R
- $R = \begin{pmatrix} 1 - 2(y^2 + z^2) & 2(xy - sz) & 2(xz + sy) \\ 2(xy + sz) & 1 - 2(x^2 + z^2) & 2(yz - sx) \\ 2(xz - sy) & 2(yz + sx) & 1 - 2(x^2 + y^2) \end{pmatrix}$

Rotations : Les quaternions

Opérations

- Vecteur
- Rotation
- Rotation sur vecteur
- Composition de rotations

Espace 3D

- $v = (v_x, v_y, v_z)$
- R(matrice 3x3)
- Rv
- $R_1 R_2$

Quaternions

- $q_v = (0, v) = (0, v_x, v_y, v_z)$
- $q = (s, x, y, z), \|q\| = 1$
- $q q_v q^*$
- $q_1 q_2$

Rotation vers quaternion

- Rotation d'axe n et d'angle θ
- $q = (\cos\left(\frac{\theta}{2}\right), n \sin\left(\frac{\theta}{2}\right))$

Interpolation de Rotations

Rotations : Les interpolations

Avec des matrices

- Trouver la rotation $\mathcal{R}(t)$ représentant la rotation intermédiaire entre $\mathcal{R}_1$ et $\mathcal{R}_2$
- $\mathcal{R}(t) = (t)\mathcal{R}_1 + (1-t)\mathcal{R}_2$ n'est pas une rotation

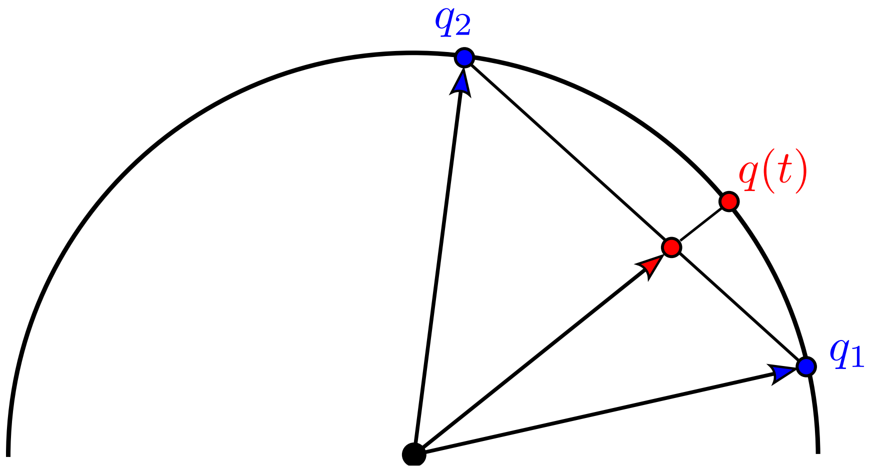
Avec Euler

- Interpolation des 3 angles
- Interpoler α, β, γ
- Pas la trajectoire la plus simple

Rotations : Les interpolations

Avec des quaternions (LERP)

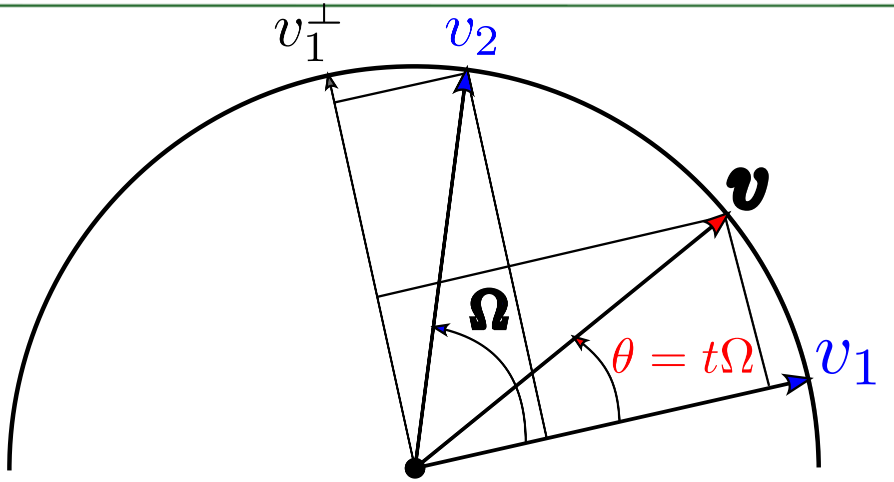
- $q(t) = \frac{(1-t)q_1 + tq_2}{\|(1-t)q_1 + tq_2\|}$
- Cercles sur sphère 4D
- Vitesse angulaire non constante



Rotations : Les interpolations

Spherical Linear Interpolation (SLERP) - vecteurs

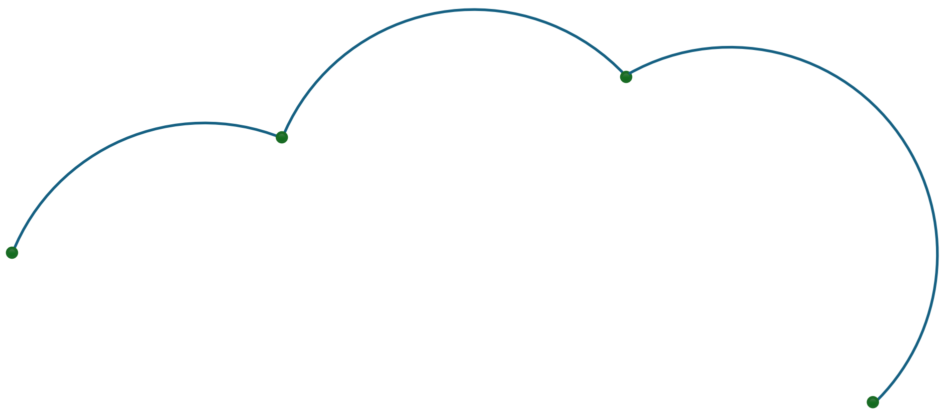
- Avec $v_1$et v_2 deux vecteurs unités
- L'interpolation sphérique en fct de t
- $v(t) = \frac{\sin((1-t)\Omega)}{\sin(\Omega)} v_1 + \frac{\sin(t\Omega)}{\sin(\Omega)} v_2$ avec $\cos\Omega = v_1 \cdot v_2$
- Intuition :
  - $v = v_1 \cos(\theta) + v_1^{\perp} \sin(\theta)$ et $v_1^{\perp} = \frac{v_2 - \cos(\Omega) v_1}{\sin(\Omega)}$



Rotations : Les interpolations

Spherical Linear Interpolation (SLERP) - quaternions

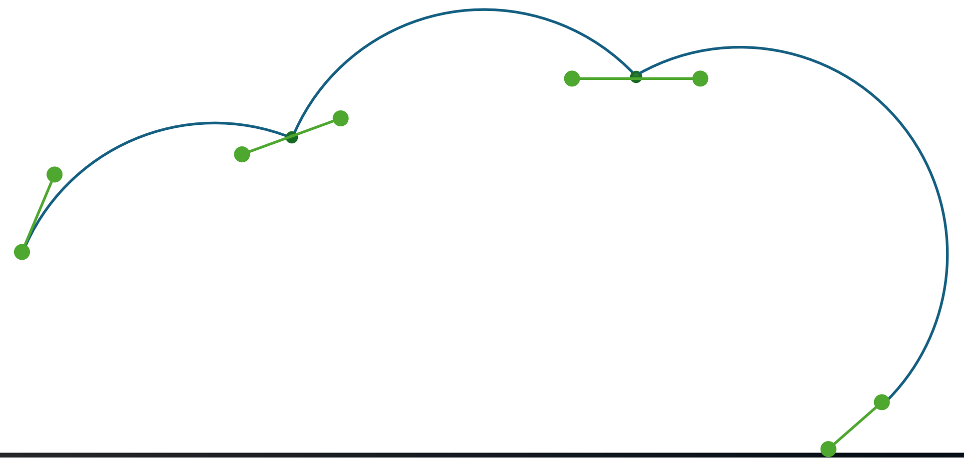
- Avec q_1 et q_2 deux quaternions unités
- L'interpolation sphérique en fct de t
- $v(t) = \frac{\sin((1-t)\Omega)}{\sin(\Omega)} q_1 + \frac{\sin(t\Omega)}{\sin(\Omega)} q_2$ avec $\cos \Omega = q_1 \cdot q_2$
- Chemin le plus court
- Vitesse angulaire constante
- Pas d'interpolation entre plus de 2 quaternions



Rotations : Les interpolations

Interpolation de plusieurs quaternions

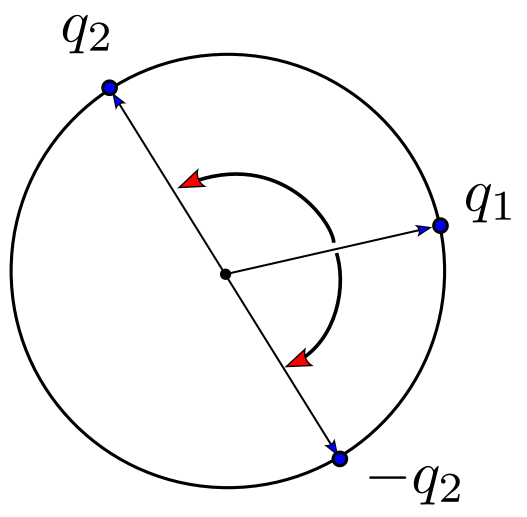
- Créer une courbe d’interpolation « Lisse »
  - Ex : Bezier cubique
- Générer des points de contrôle (avec l’interpolation linéaire sphérique)
  - Avant/après chaque point
- Algorithme de DeCasteljeau
- Succession d’interpolations sphériques (slerp(slerp(...))



Rotations : Les interpolations

Attention à la négation

- $+q$ et -q correspondent à la même rotation
- Mais un chemin différent dans l'espace 4D
- Le chemin $q_1 \rightarrow -q_2$ est plus court que $q_1 \rightarrow q_2$ quand $q_1 \cdot q_2 < 0$



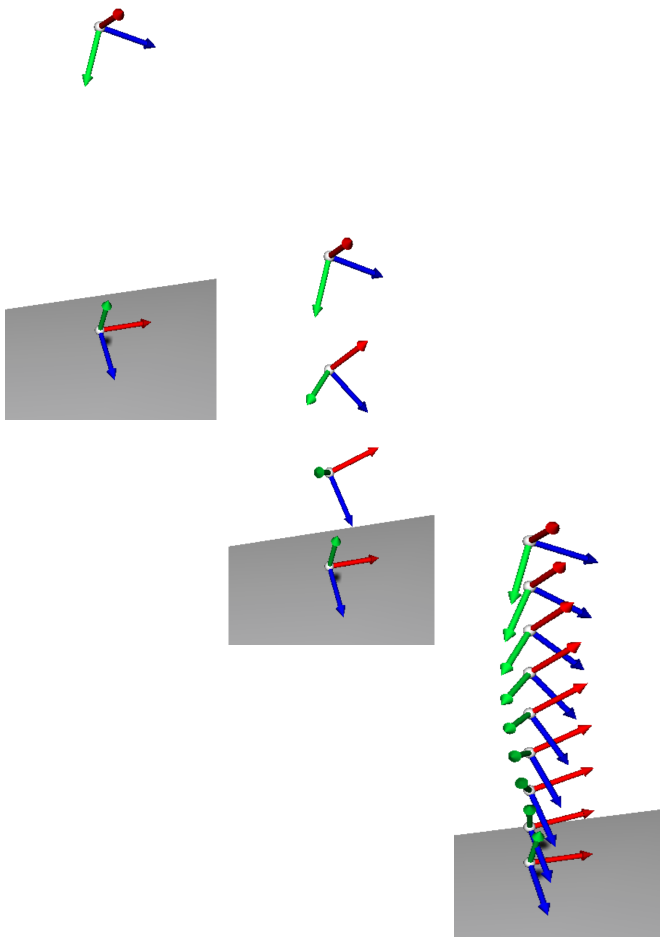
Les interpolations

Interpoler un déplacement rigide

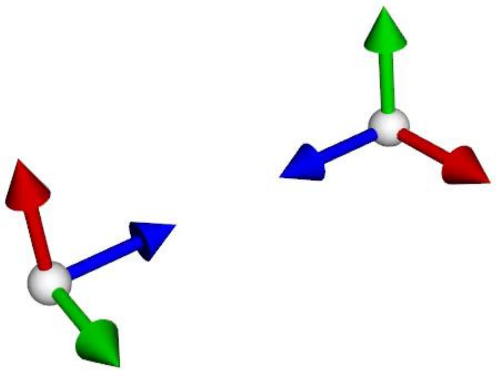
- Les objets 3d ont une position et une orientation
- Soit deux keyframes $(p_{1,}r_1)$ et $(p_{2,}r_2)$

Séparer positions p et orientations r

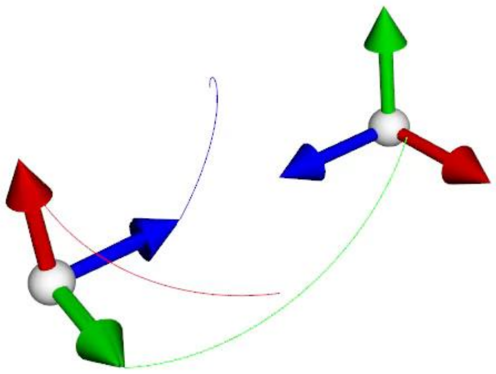
- Interpolation des positions (linéaire ou non)
  - Ex: $p(t) = (1 - t)p_1 + t\ p_2$
- Interpolation des rotations
  - Convertir $(r_{1,}r_2) \rightarrow (q_{1,}q_2)$
  - $q(t) = \ SLERP(q_1, q_2, t)$
  - Convertir $q(t) \rightarrow r(t)$



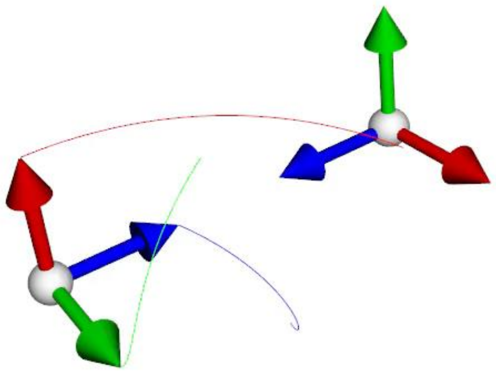
Rotations : Les interpolations



Interpolation de matrices



Interpolation d'angles d'Euler



Interpolation de quaternions