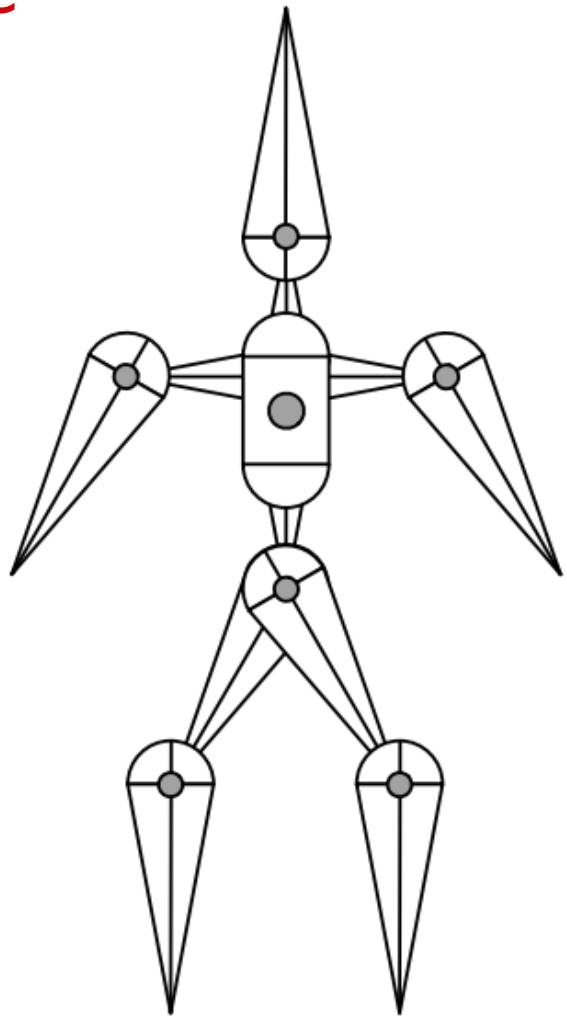
Plan

1. Introduction
2. **Cinématique**
   - Interpolation positions / vitesses / rotations
   - **Modélisation hiérarchique**
   - **Cinématique directe**
   - **Cinématique inverse**
3. Déformation de surfaces

Modélisation hiérarchique

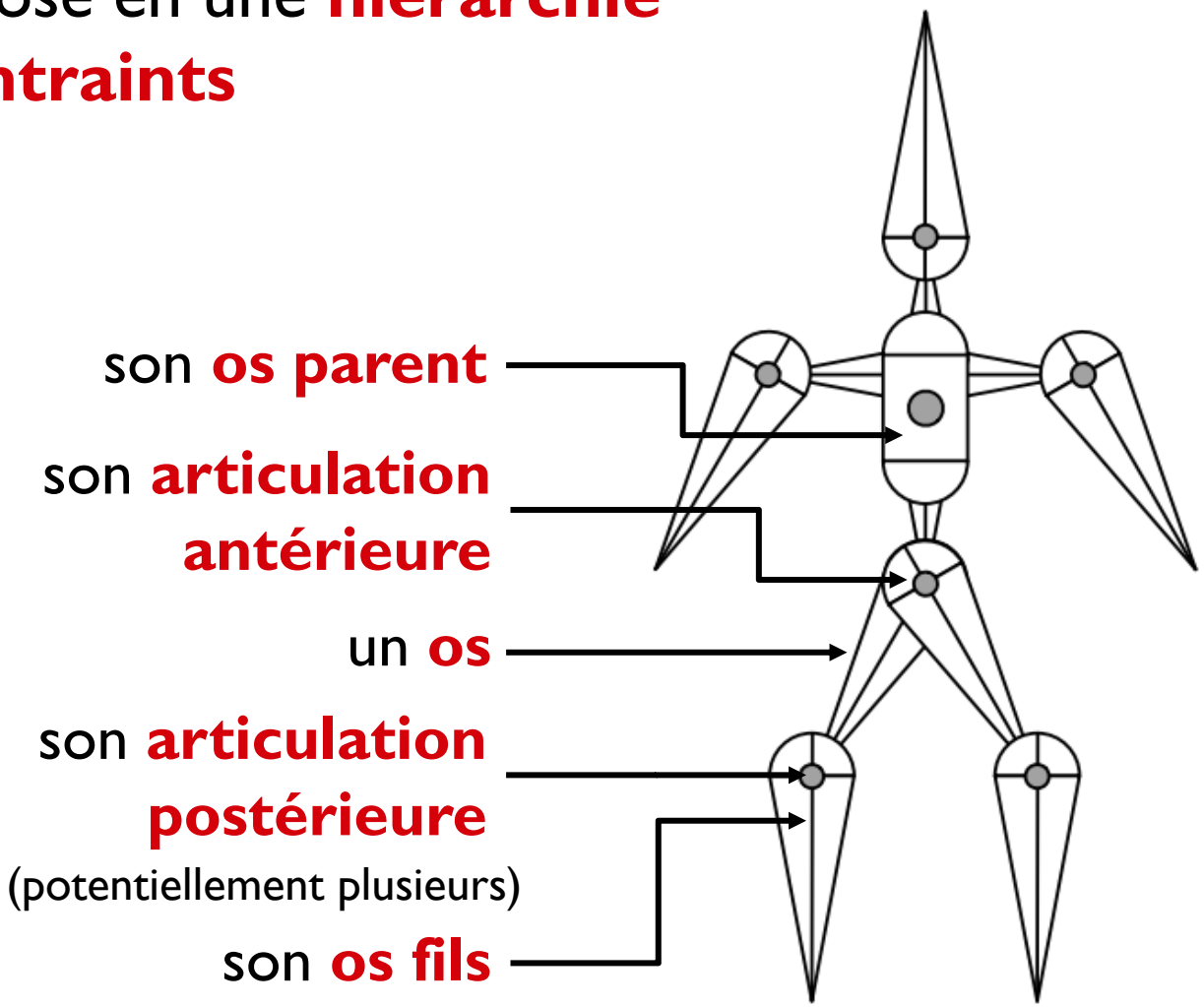
Modèle décomposé en une **hiérarchie de repères contraints**

Pas de géométrie, que des solides appelés **os** (***bones***) et des liaisons appelés **articulations** (***joins***)



Modélisation hiérarchique

Modèle décomposé en une **hiérarchie de repères contraints**



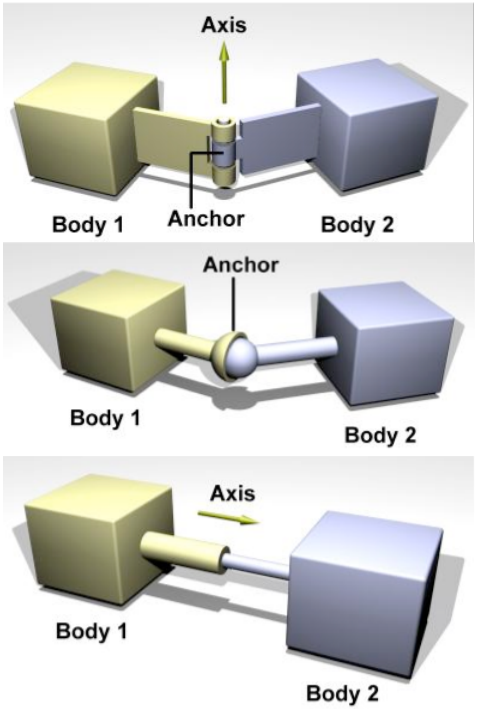
Articulations

En général **6 degrés de liberté (DOF)** :

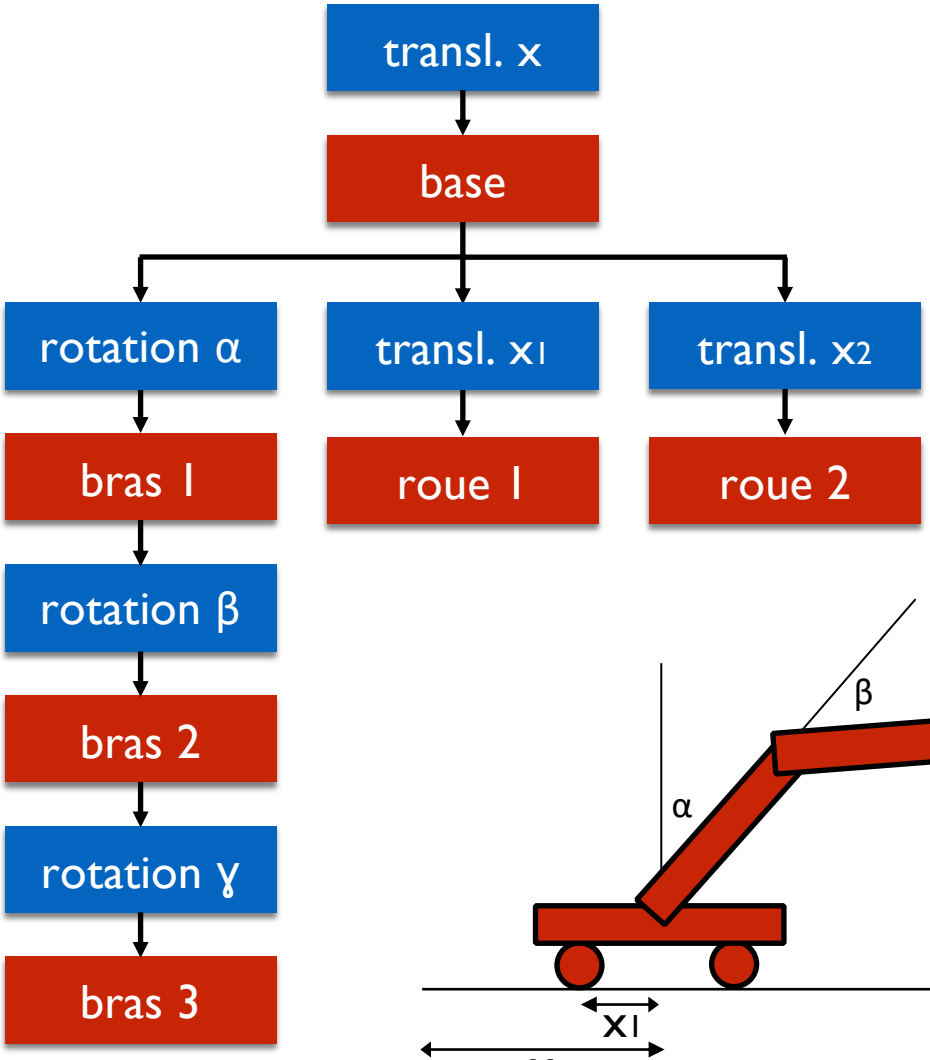
- 3 translations
- 3 rotations

En pratique, souvent des **contraintes** :

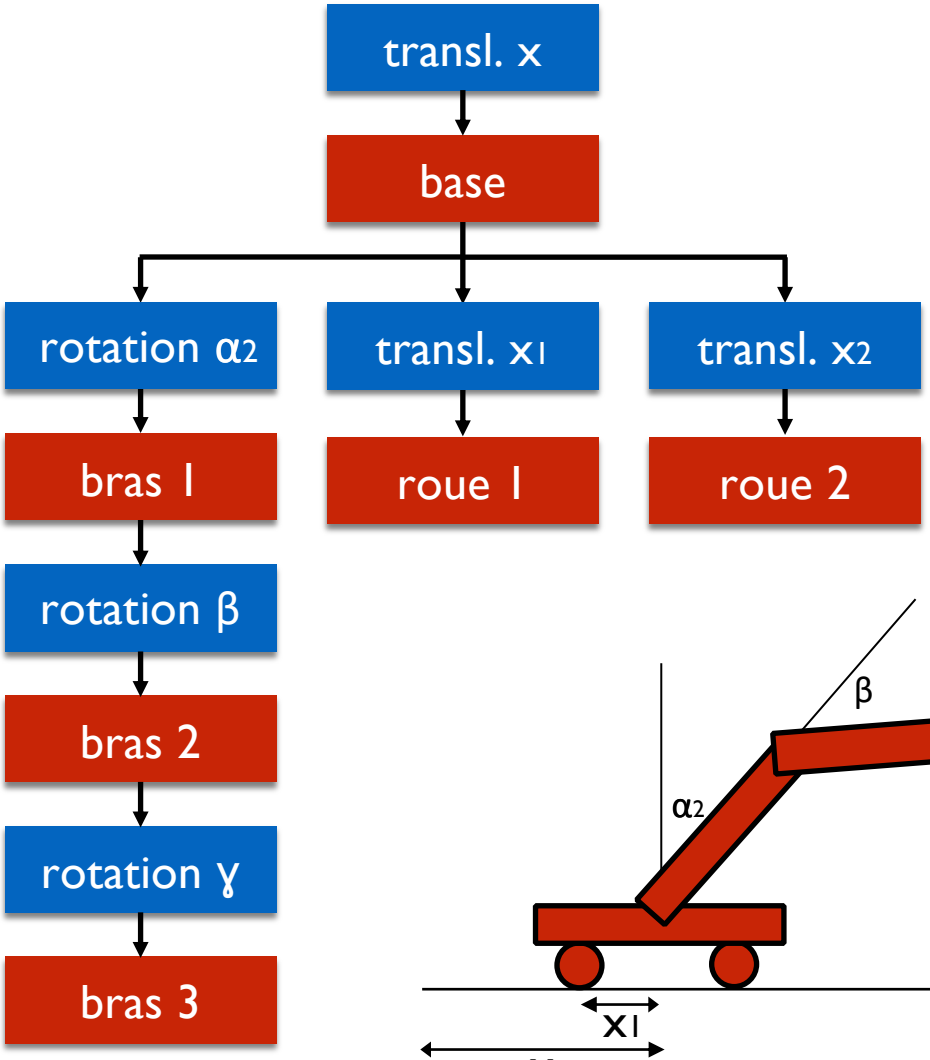
- ***Pin*** – rotation autour d'un axe (1 DOF)
- ***Ball*** – rotation arbitraire (3 DOF)
- ***Prism*** – translation selon un axe (1 DOF)



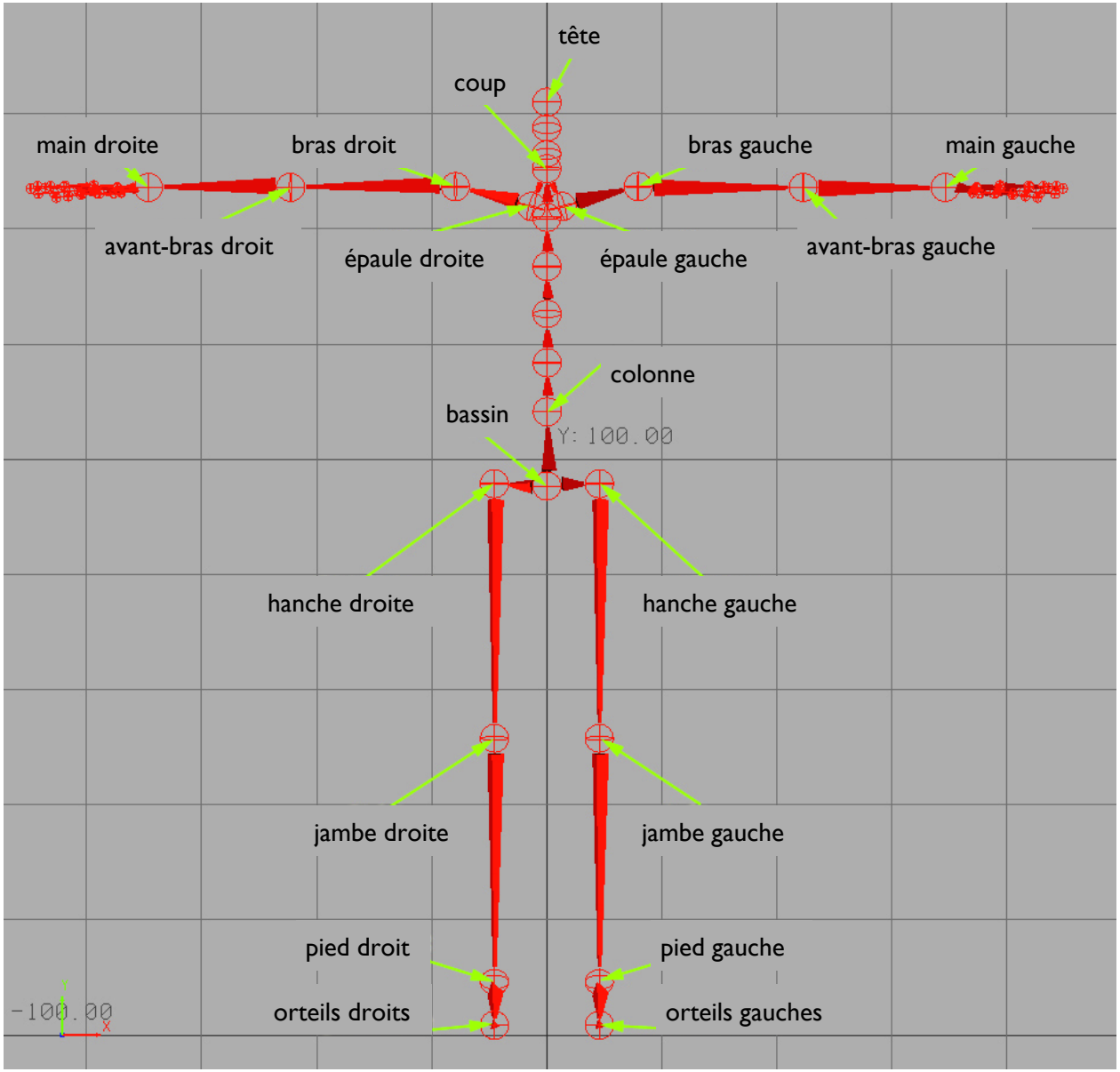
Chaîne articulée



Chaîne articulée

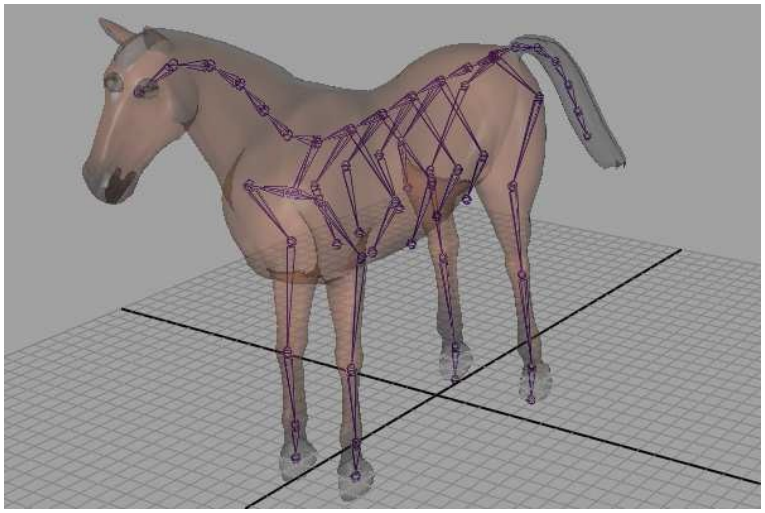


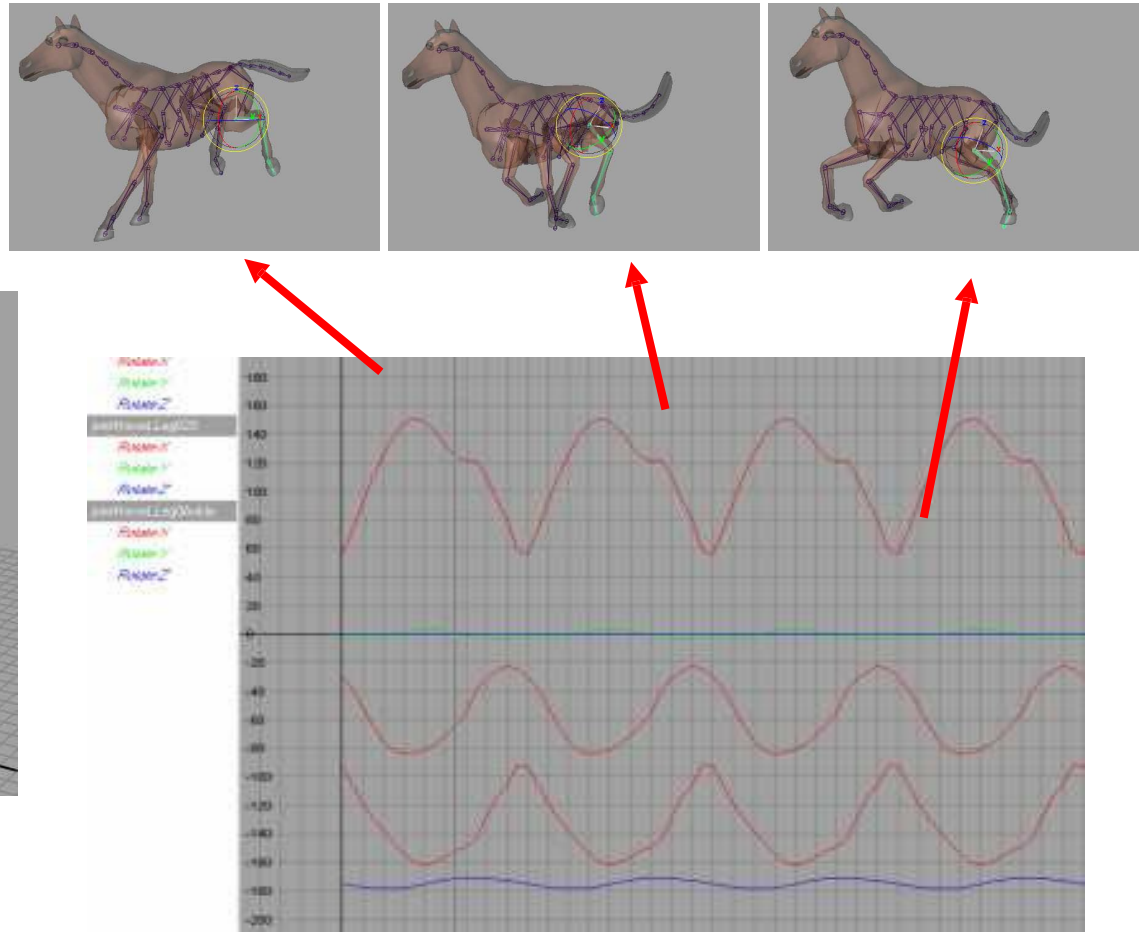
Squelette d'animation



Animation 3D

Squelette + interpolation selon une courbe





Cinématique

Mouvement non-contraint

Hocher la tête, faire un signe de la main…

$\Longrightarrow$ Cinématique directe (FK)

Mouvement contraint

Attraper un objet, marcher sur le sol…

$\Longrightarrow$ Cinématique inverse (IK)

Cinématique directe

Entrées : configuration de la chaine = valeurs des paramètres des articulations à un temps donnée :

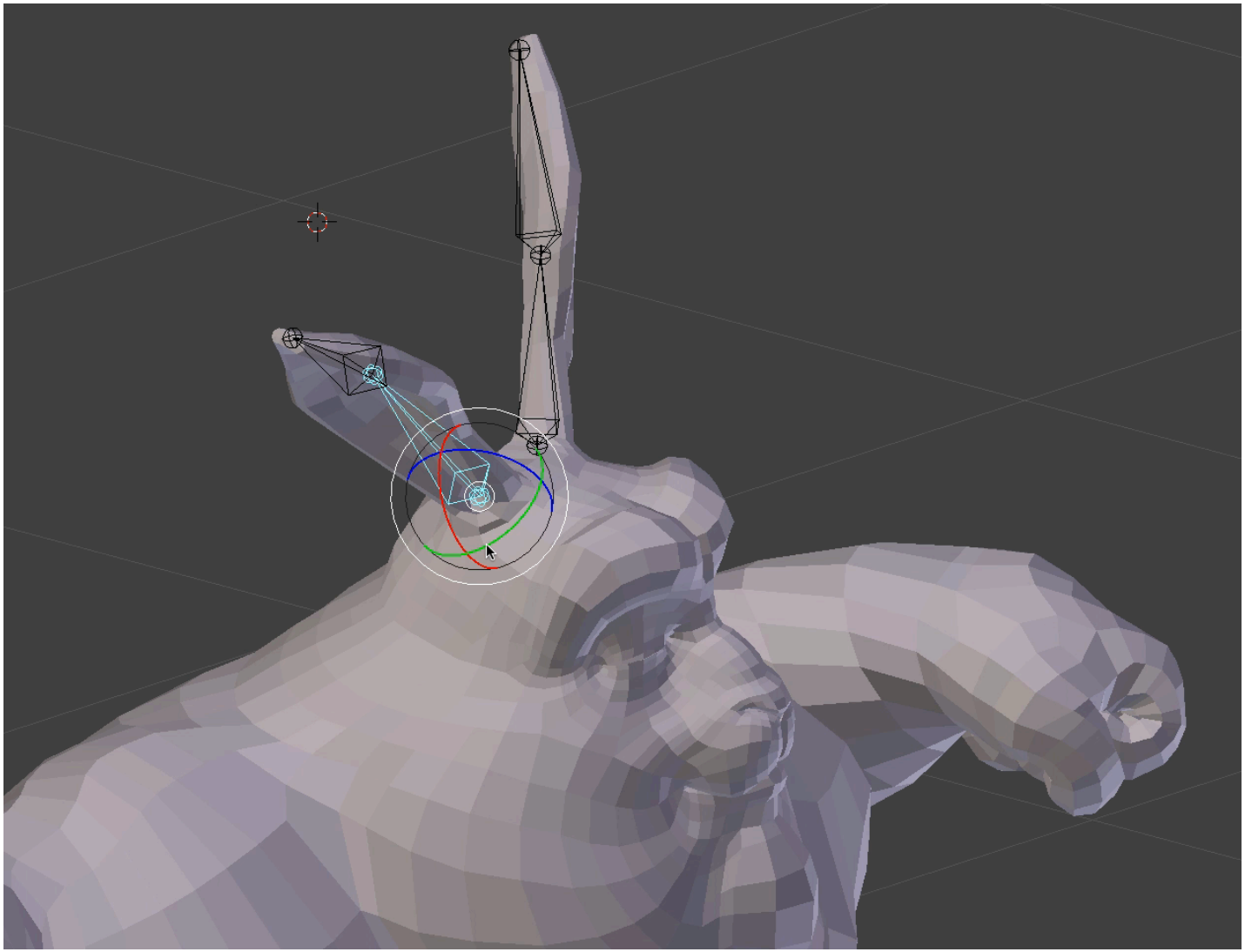
$$\mathbf{q} = (\theta_1, \theta_2, \theta_3, ..., d_1, d_2, ...)$$

Sortie : position $\mathbf{p}$ d'une extrémité de la chaîne

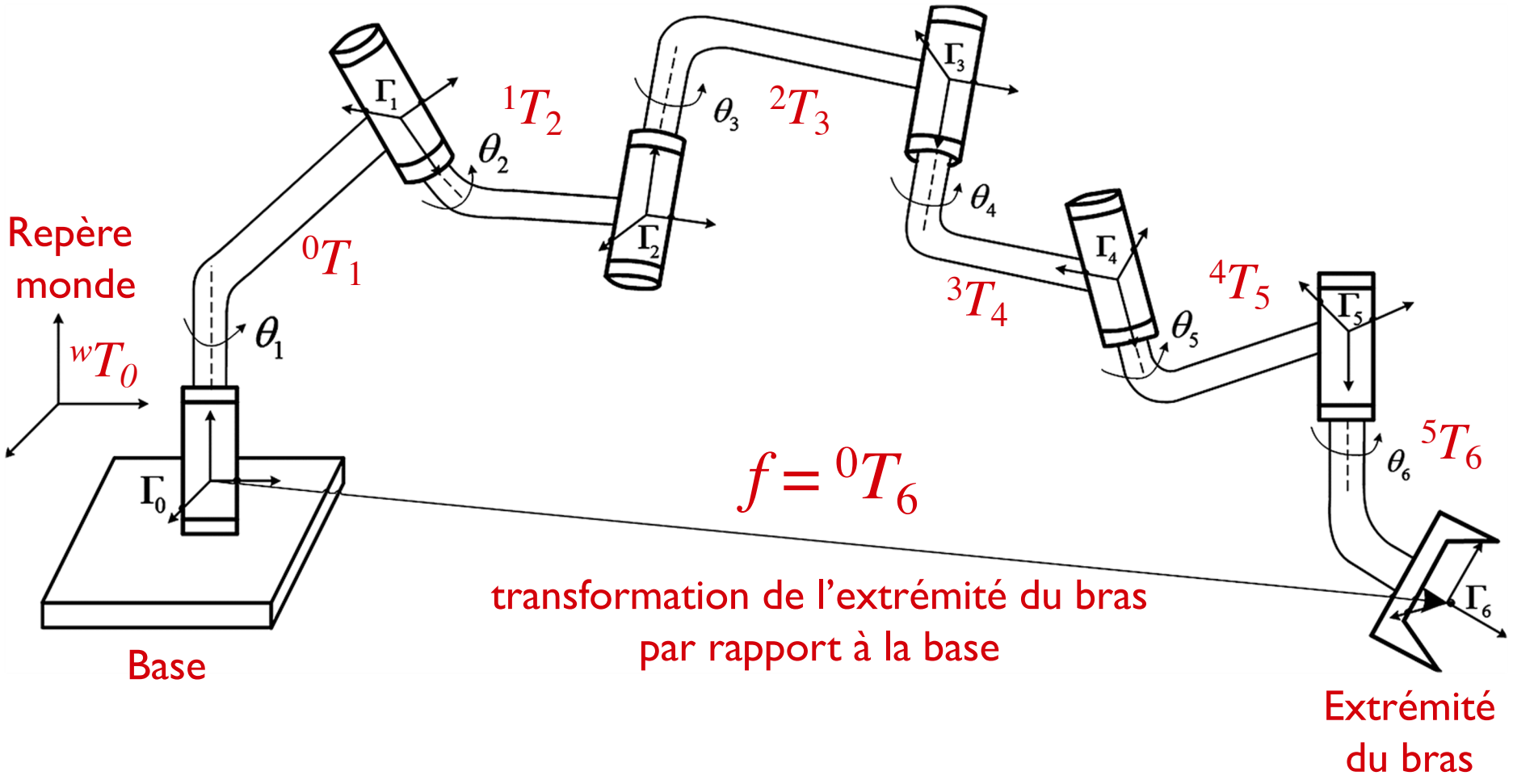
(coordonnées cartésiennes + orientation, 3 ou 6D)

On **calcule** : $\mathbf{p} = f(\mathbf{q})$

Cinématique directe

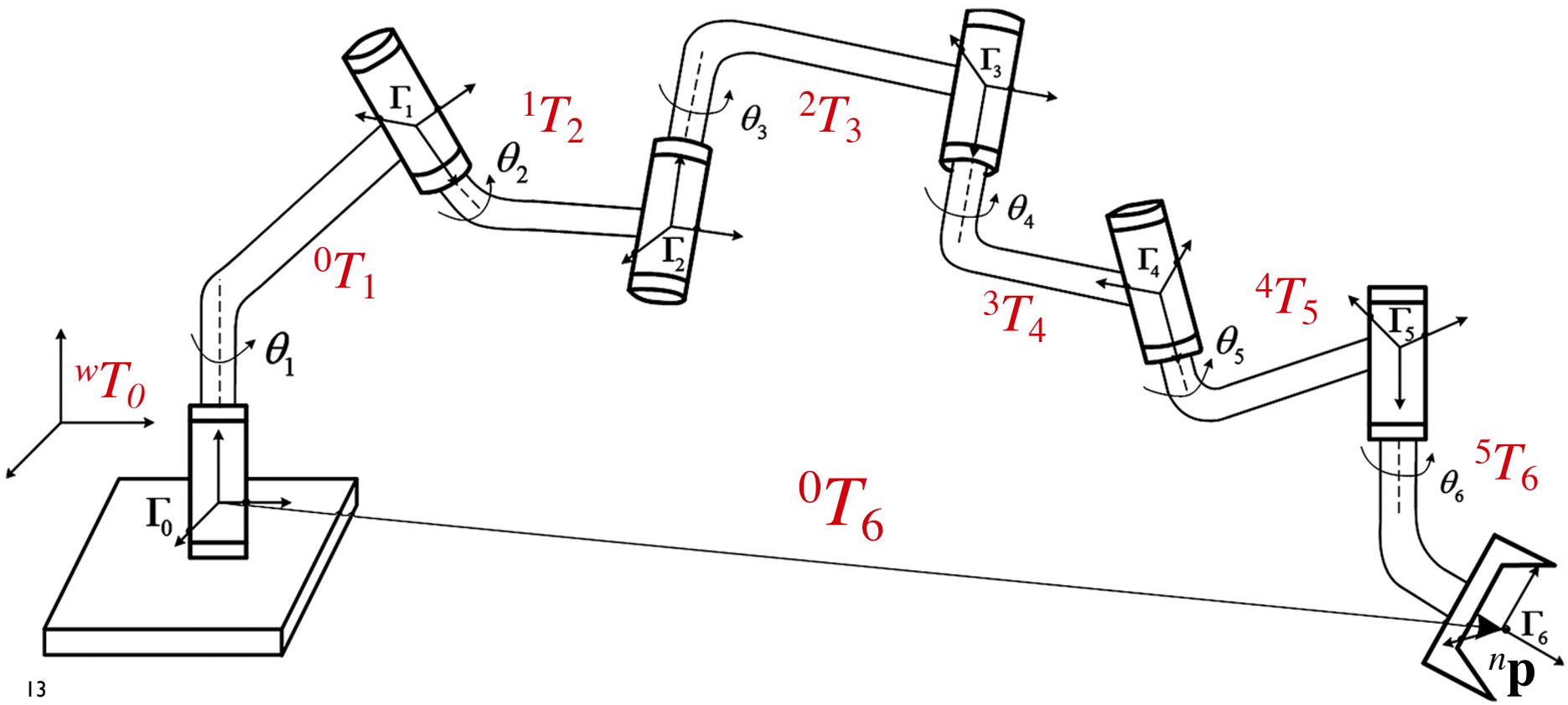


Que vaut f ?


Repère monde
$^wT_0$
$^0T_1$
$^1T_2$
$^2T_3$
$^3T_4$
$^4T_5$
$^5T_6$
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6
Γ_0
Γ_1
Γ_2
Γ_3
Γ_4
Γ_5
Γ_6
$f = {}^0T_6$
transformation de l'extrémité du bras par rapport à la base
Base
Extrémité du bras

Que vaut f ?

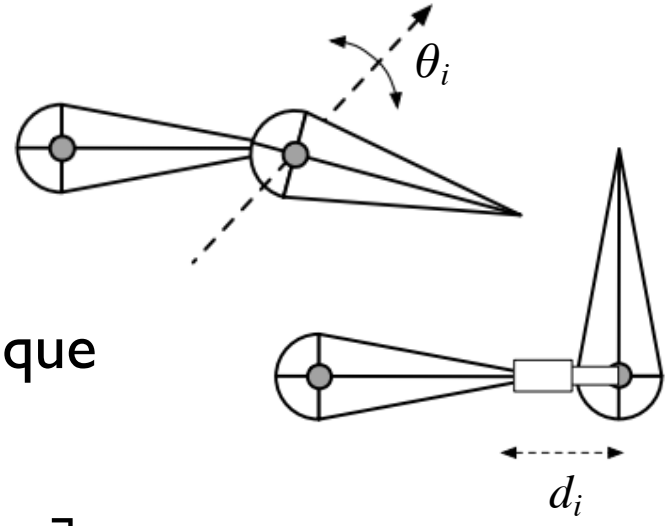
$$\mathbf{p} = {}^0\mathbf{p}_n = \underbrace{{}^0T_1\ {}^1T_2 \cdots {}^{n-1}T_n}_{= {}^0T_n = f}\ {}^n\mathbf{p}$$



Que vaut ${}^0T_n$?

Configuration

$$q_i = \begin{cases} \theta_i & \text{pour une liaison pivot} \\ d_i & \text{pour une liaison prismatique} \end{cases}$$



en matrice homogène :

$$A_i = \begin{bmatrix} {}^{i-1}R_i & {}^{i-1}\mathbf{o}_i \\ \mathbf{0} & 1 \end{bmatrix}$$

Transformation entre les repères i et j :

$${}^iT_j = A_{i+1} \dots A_j = \begin{bmatrix} {}^iR_j & {}^i\mathbf{o}_j \\ \mathbf{0} & 1 \end{bmatrix}$$

avec ${}^iR_j = {}^iR_{i+1} \dots {}^{j-1}R_j$ et ${}^i\mathbf{o}_j = {}^i\mathbf{o}_{j-1} + {}^iR_{j-1}\,{}^{j-1}\mathbf{o}_j$

Cinématique directe

Limitations

- Contrôle des extrémités difficile
  (outils pour changer l'origine de la chaîne articulée)
- Accumulation des erreurs

Cinématique inverse

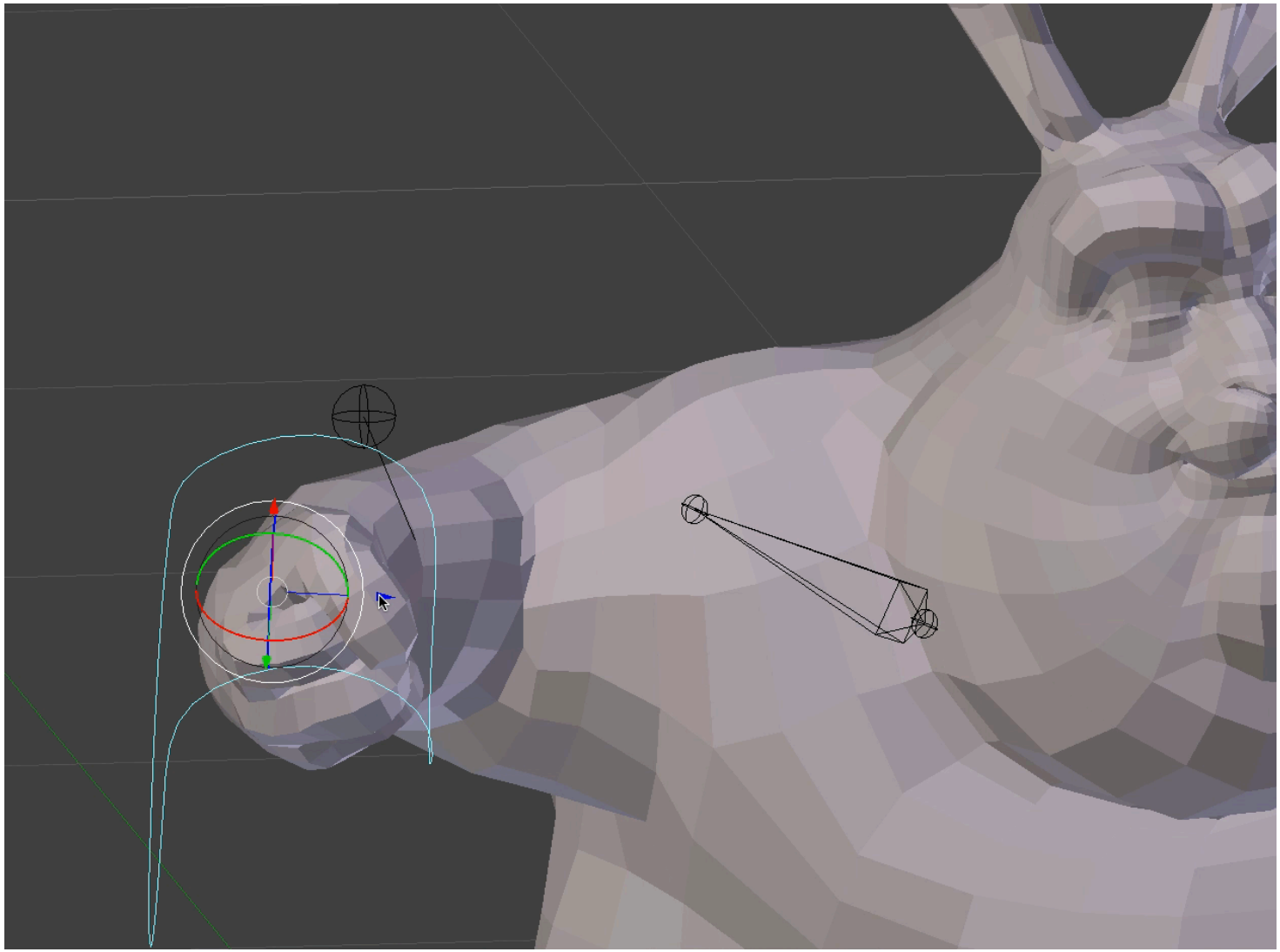
Entrée : position à atteindre $\mathbf{p}$

Sortie : valeurs des paramètres des articulations

$$\mathbf{q} = (\theta_1, \theta_2, \theta_3, ..., d_1, d_2, ...)$$

On veut **trouver q** tel que $f(\mathbf{q}) = \mathbf{p}$ c'est à dire $\mathbf{q} = f^{-1}(\mathbf{p})$

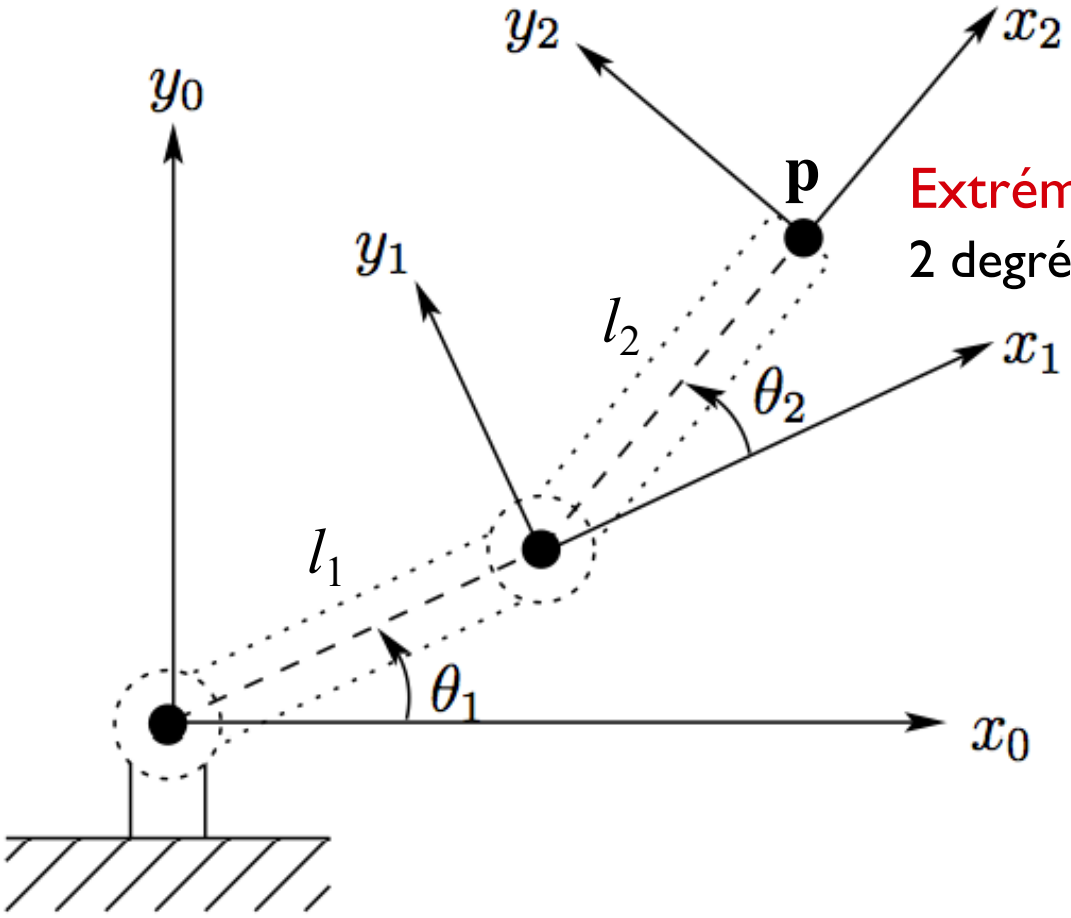
Cinématique inverse



Bras 2D

Configuration $\mathbf{q} = (\theta_1, \theta_2)$

2 degrés de liberté angulaires



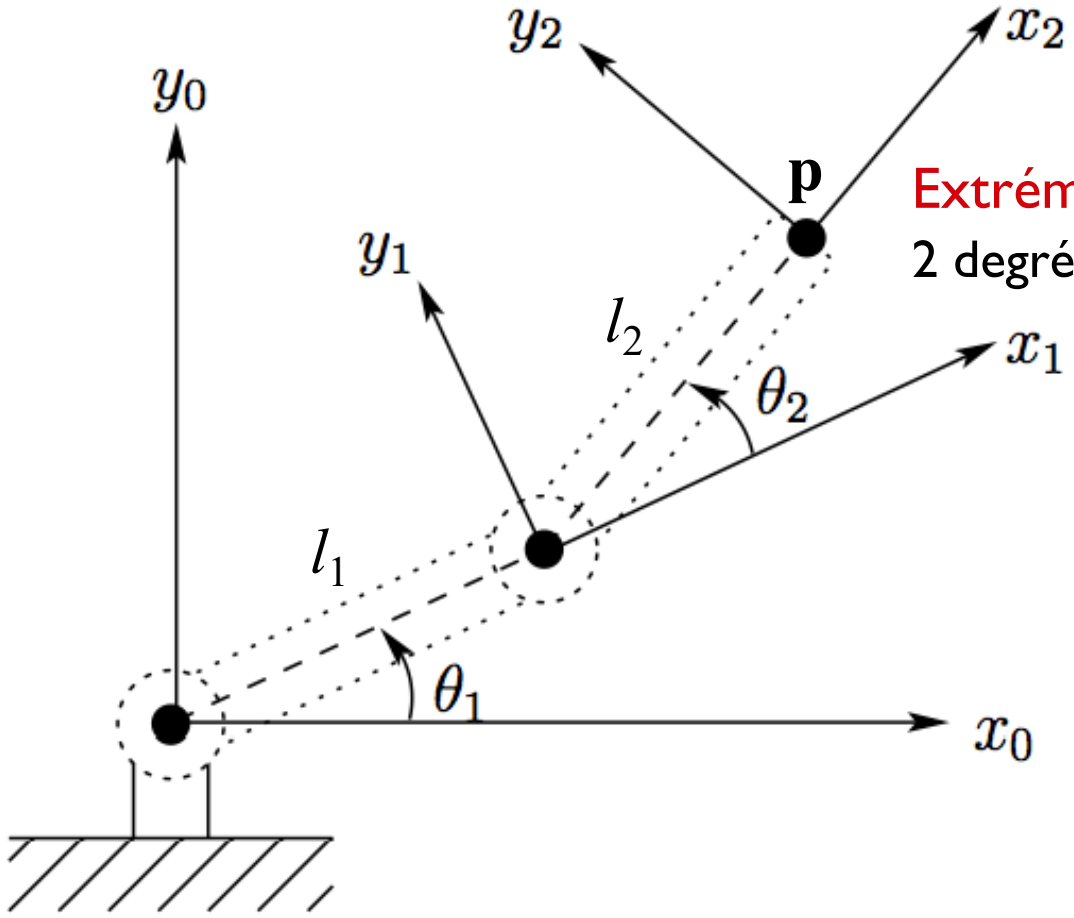
Extrémité du bras $\mathbf{p} = (p_x, p_y)$

2 degrés de liberté cartésiennes

Bras 2D – cinématique directe

Configuration $\mathbf{q} = (\theta_1, \theta_2)$

2 degrés de liberté angulaires



Extrémité du bras $\mathbf{p} = (p_x, p_y)$

2 degrés de liberté cartésiennes

$$\mathbf{p} = f(\mathbf{q}) = {}^0T_1 \, {}^1T_2 \, {}^2\mathbf{p}$$

avec ${}^0T_1 = ?$

${}^1T_2 = ?$

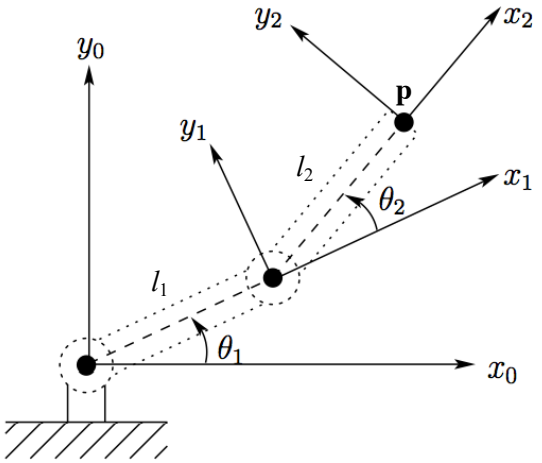
Bras 2D – cinématique directe

$$ {}^0T_1 = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & l_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & l_1\sin\theta_1 \\ 0 & 0 & 1 \end{pmatrix} \qquad {}^1T_2 = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & l_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & l_2\sin\theta_2 \\ 0 & 0 & 1 \end{pmatrix} $$

D'où :

$p_x = ?$

$p_y = ?$



Rappel formules de trigo :

$$\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y)$$
$$\sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y)$$

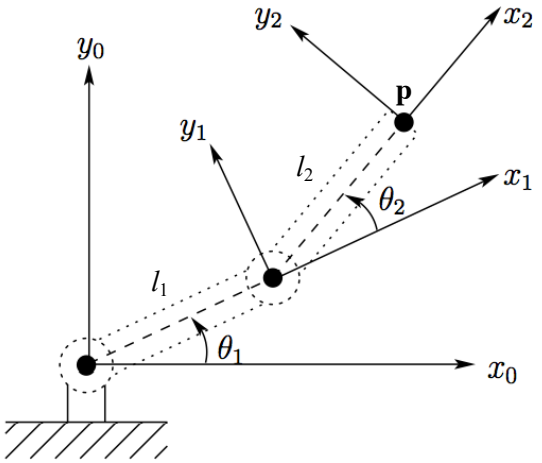
Bras 2D – cinématique directe

$$ {}^0T_1 = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & l_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & l_1\sin\theta_1 \\ 0 & 0 & 1 \end{pmatrix} \quad {}^1T_2 = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & l_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & l_2\sin\theta_2 \\ 0 & 0 & 1 \end{pmatrix} $$

D'où :

$$ p_x = l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2) $$

$$ p_y = l_1\sin\theta_1 + l_2\sin(\theta_1+\theta_2) $$



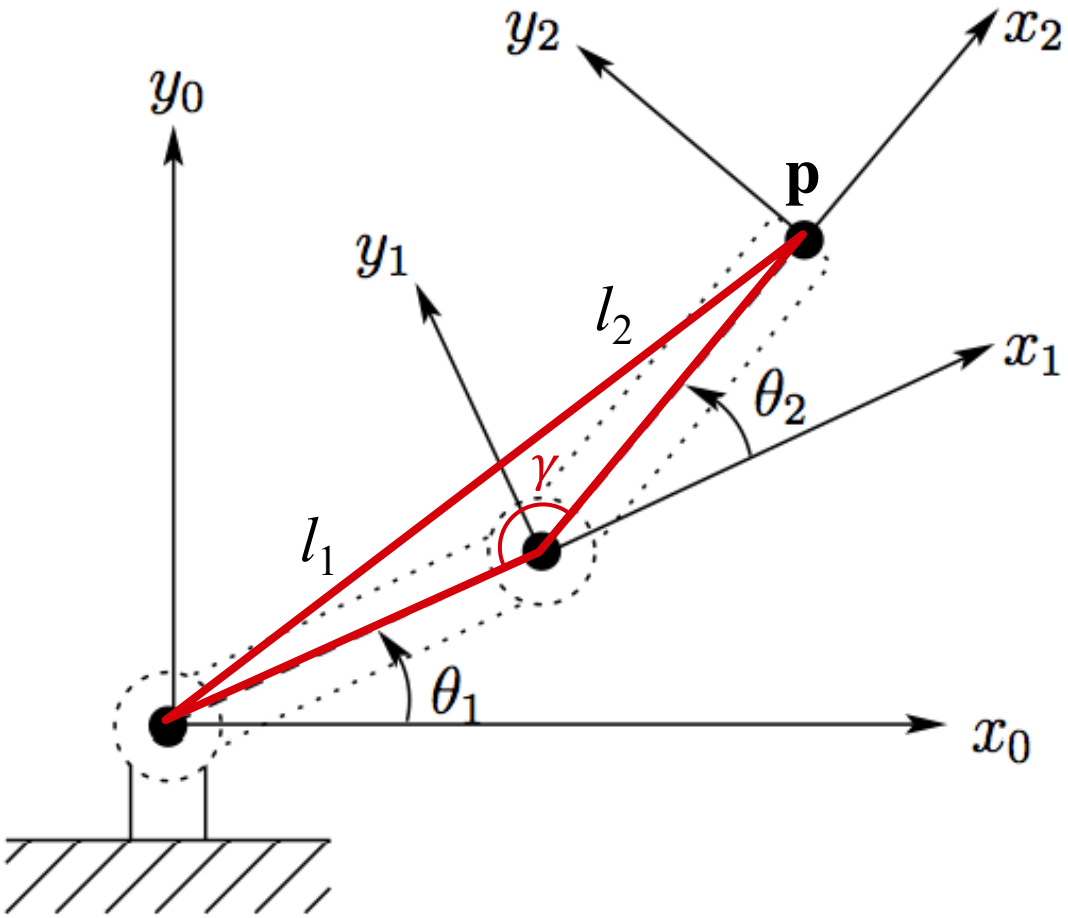
Rappel formules de trigo :

$$ \cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y) $$

$$ \sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y) $$

Bras 2D – cinématique inverse

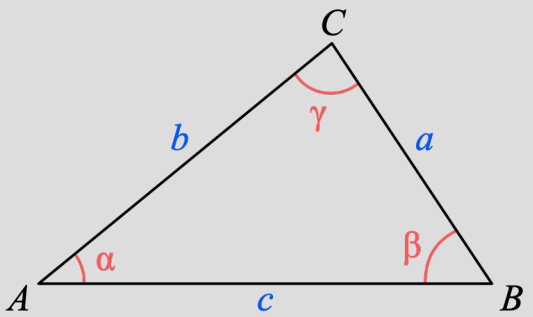
Configuration $\mathbf{q} = (\theta_1, \theta_2) = f^{-1}(\mathbf{p})$



1. Cherchons θ_2

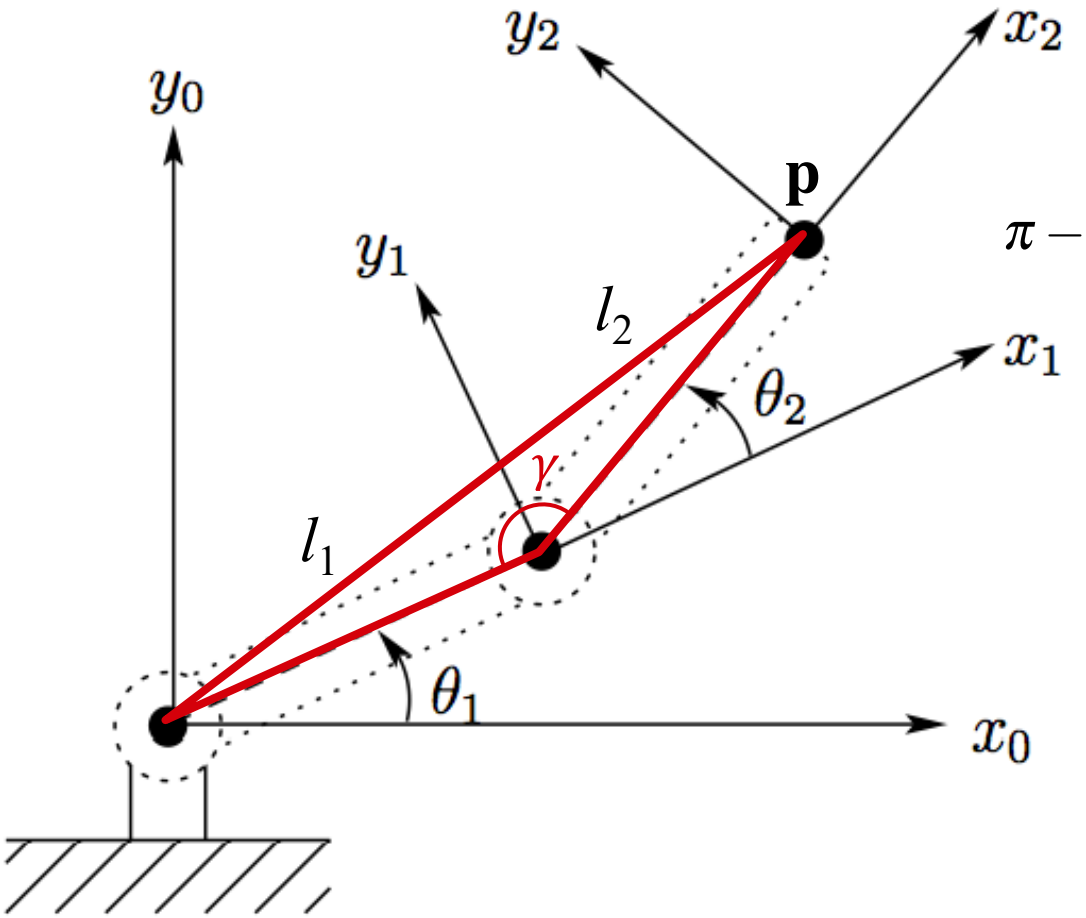
Rappel théorème d'Al-Kashi :

$$\gamma = \cos^{-1} \frac{a^2 + b^2 - c^2}{2ab}$$



Bras 2D – cinématique inverse

Configuration $\mathbf{q} = (\theta_1, \theta_2) = f^{-1}(\mathbf{p})$

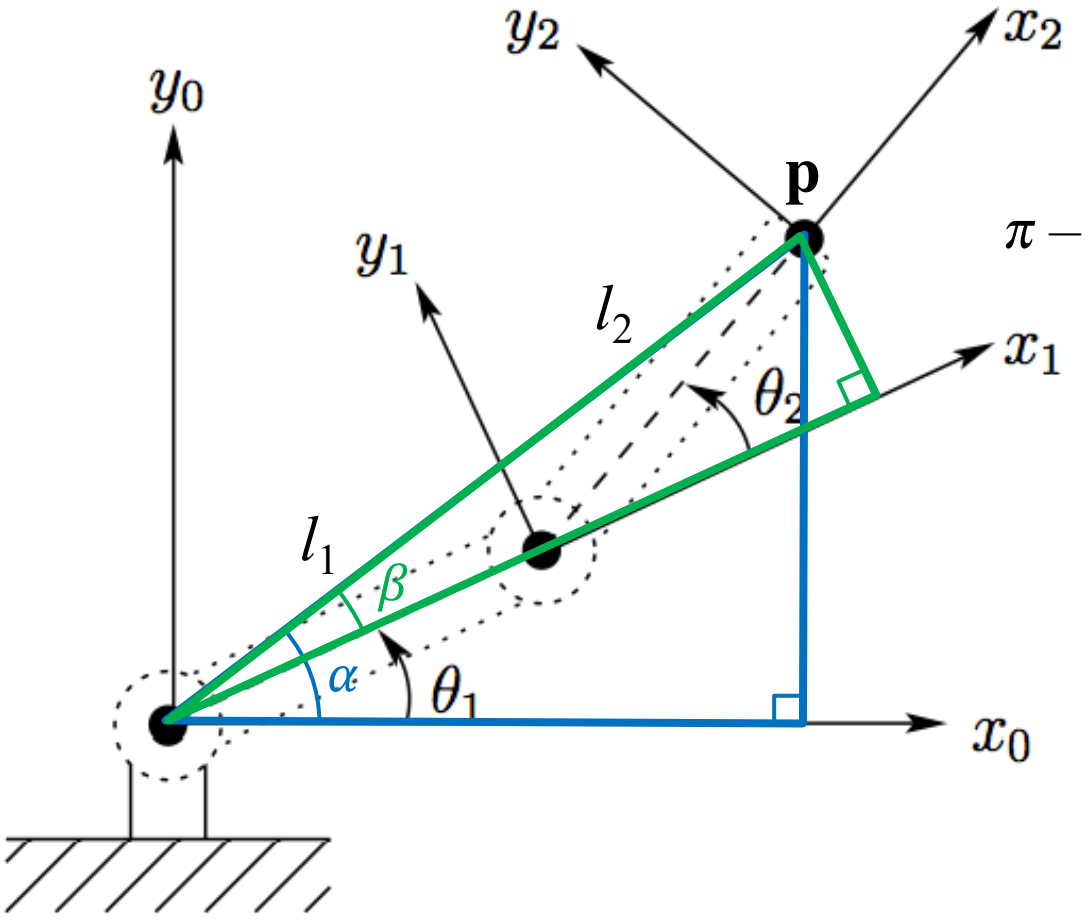


1. Cherchons θ_2

$$\pi - \theta_2 = \cos^{-1} \frac{l_1^2 + l_2^2 - (p_x^2 + p_y^2)}{2\, l_1\, l_2}$$

Bras 2D – cinématique inverse

Configuration $\mathbf{q} = (\theta_1, \theta_2) = f^{-1}(\mathbf{p})$



1. Cherchons θ_2

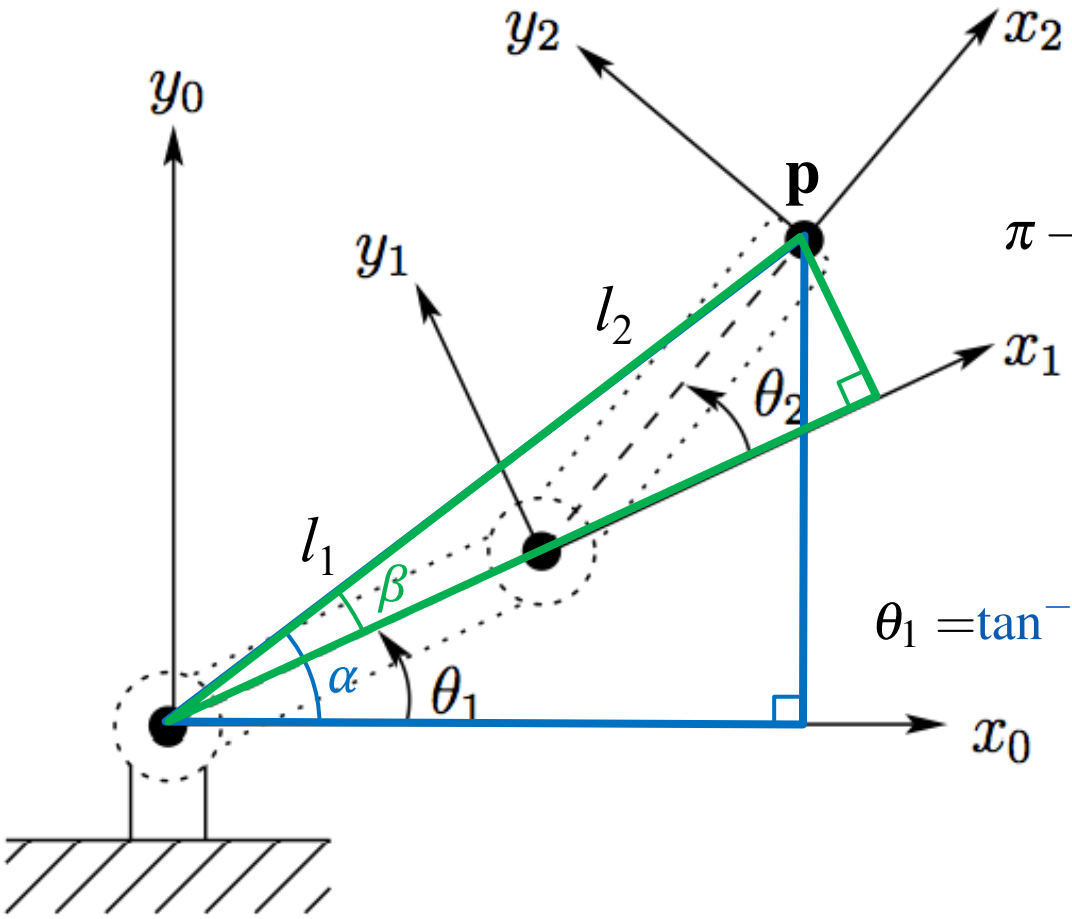
$$\pi - \theta_2 = \cos^{-1} \frac{l_1^2 + l_2^2 - (p_x^2 + p_y^2)}{2\, l_1\, l_2}$$

2. Cherchons θ_1

$\theta_1 = \alpha - \beta$

Bras 2D – cinématique inverse

Configuration $\mathbf{q} = (\theta_1, \theta_2) = f^{-1}(\mathbf{p})$



1. Cherchons θ_2

$$\pi - \theta_2 = \cos^{-1} \frac{l_1^2 + l_2^2 - (p_x^2 + p_y^2)}{2 l_1 l_2}$$

2. Cherchons θ_1

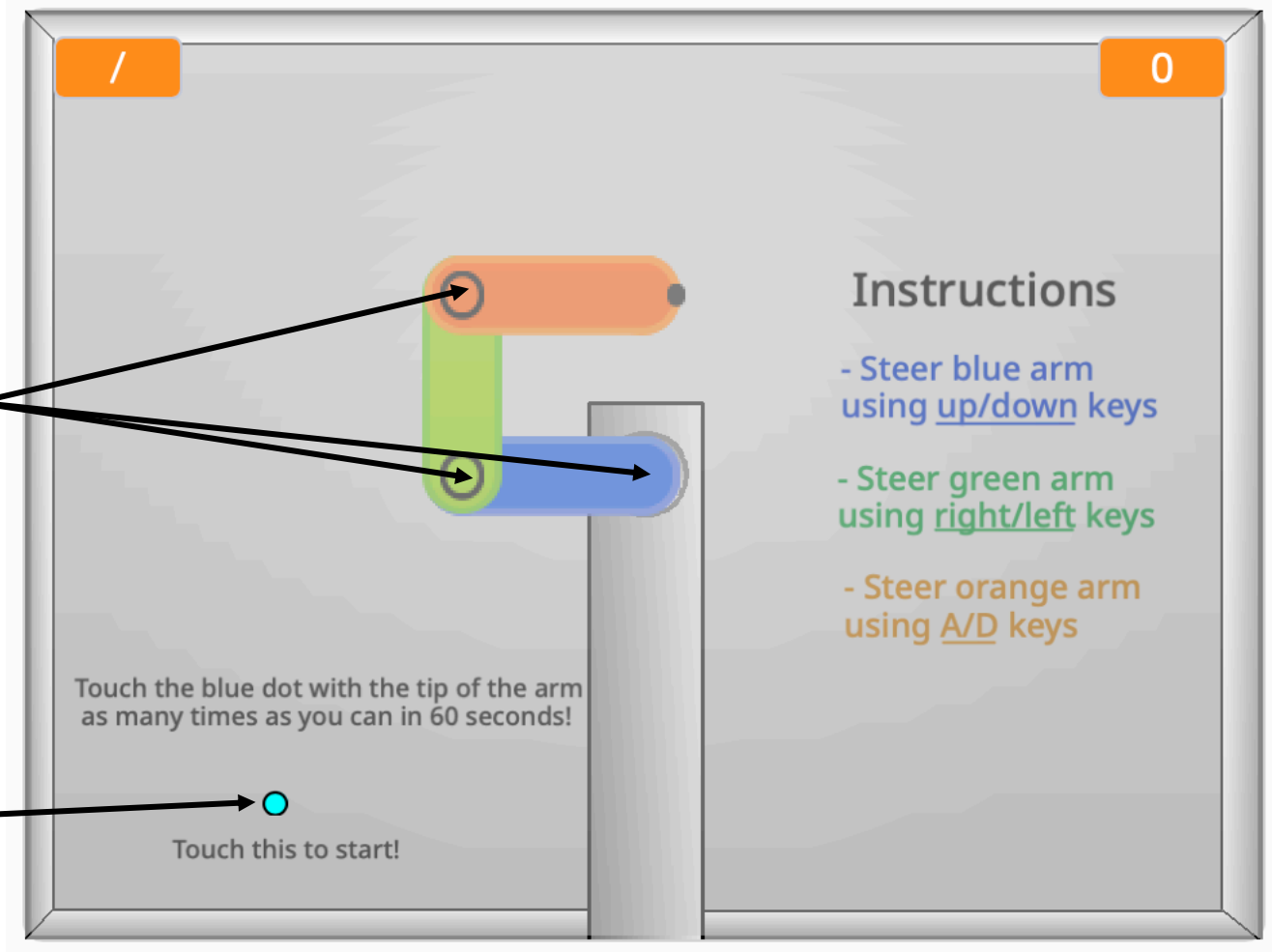
$$\theta_1 = \alpha - \beta$$

$$\theta_1 = \tan^{-1}\left(\frac{p_y}{p_x}\right) - \tan^{-1}\left(\frac{l_2 \sin\theta_2}{l_1 + l_2 \cos\theta_2}\right)$$

Bras 2D – 3 articulations

3 angles inconnues

contrainte de position 2D

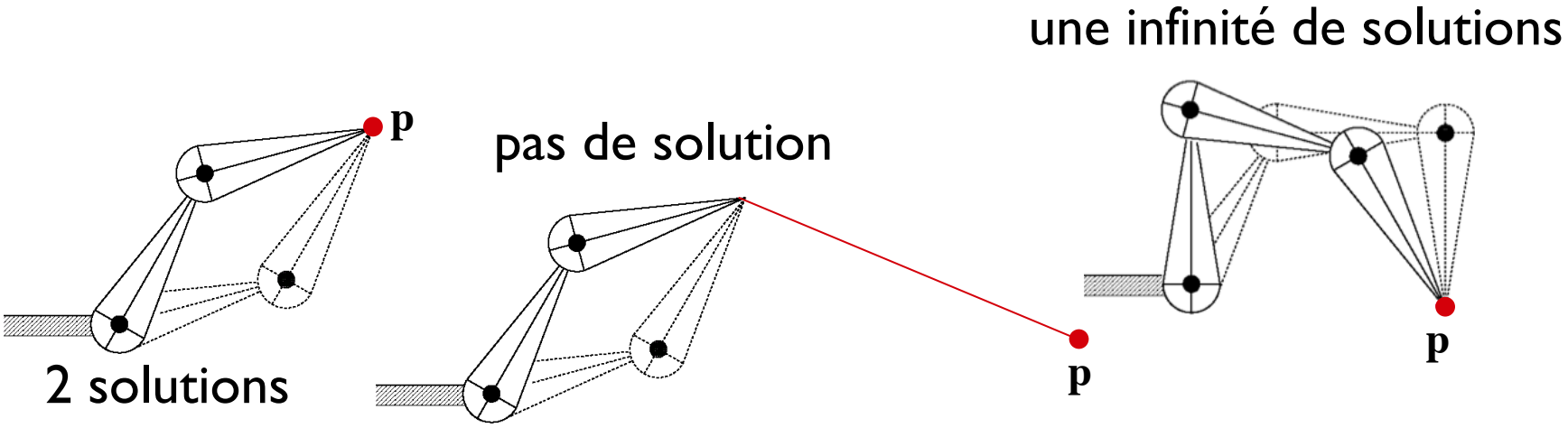


http://scratch.mit.edu/projects/10607750

Cinématique inverse

Difficultés :

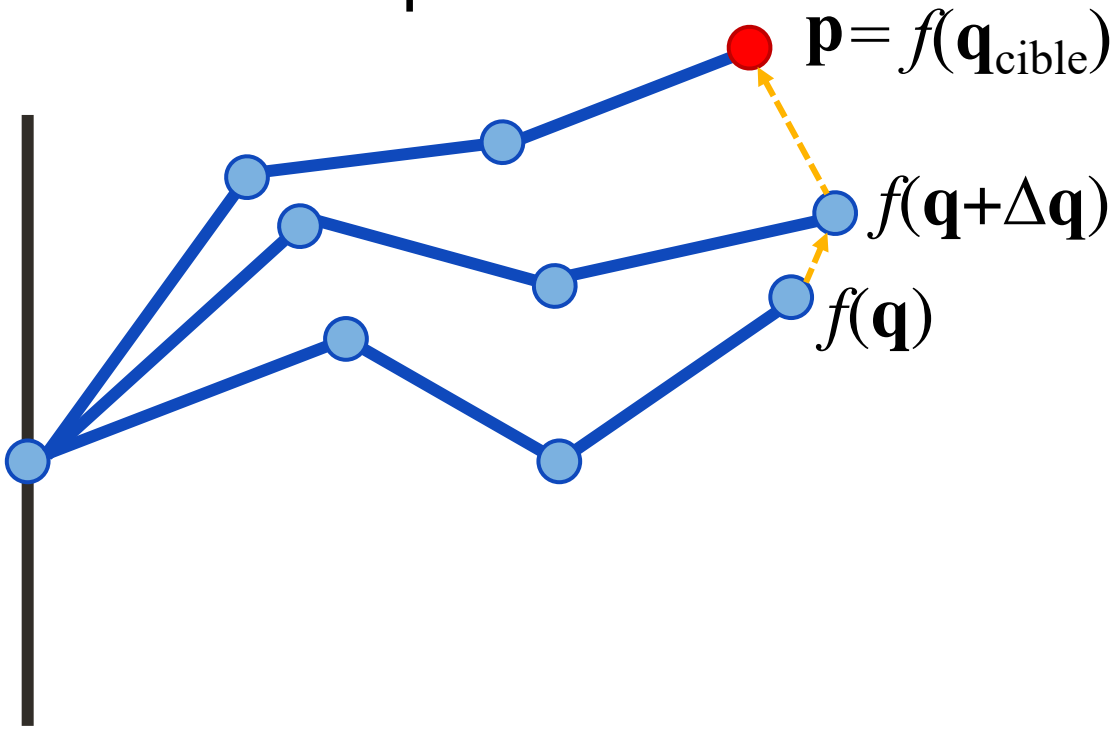
- Solution analytique seulement pour 2 à 3 rotations
- Non-unicité de la solution



$\Longrightarrow$ rechercher **itérativement** une solution

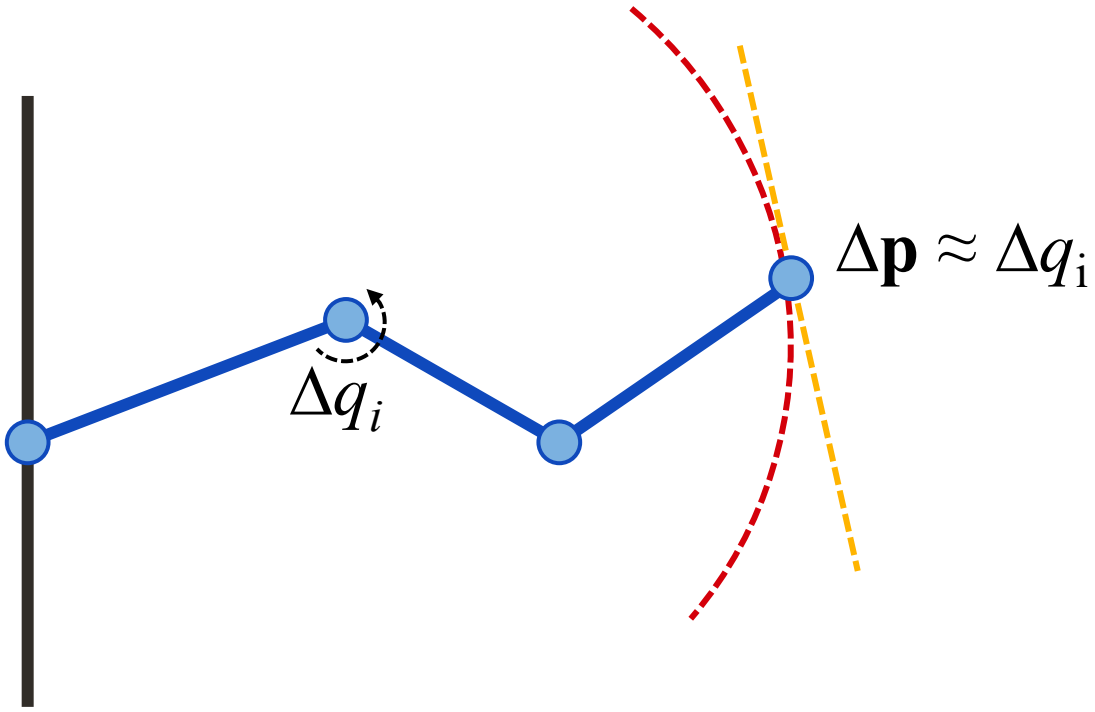
Solutions itératives

Minimiser itérativement l'erreur $e(\mathbf{q})$ entre la position courante et la position cible



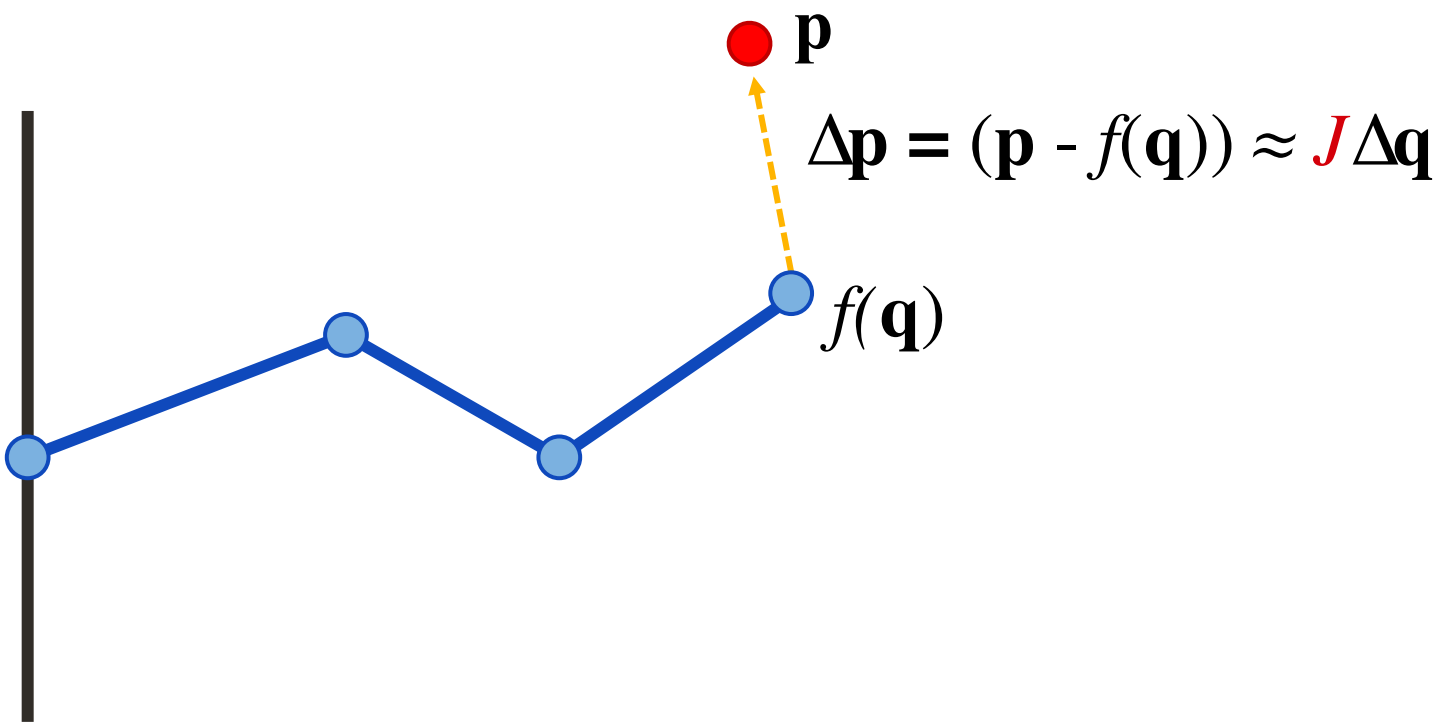
Problème : optimisation **non-linéaire** à cause des rotations

Approximation linéaire de f



…par son **Jacobien géométrique**

Approximation linéaire de f



…par son **Jacobien géométrique**

Jacobien géométrique

Si $\Delta\mathbf{q}$ est petit, développement de Taylor :

- En 1D :

$$f(\mathbf{q} + \Delta\mathbf{q}) = f(\mathbf{q}) + f'(\mathbf{q})\,\Delta\mathbf{q} + O(\Delta\mathbf{q}^2)$$

- En dimensions supérieures :

$$f(\mathbf{q} + \Delta\mathbf{q}) = f(\mathbf{q}) + J(\mathbf{q})\,\Delta\mathbf{q} + O(\Delta\mathbf{q}^2)$$

$J(\mathbf{q})$ **matrice** (6xn) **des dérivées partielles** de f :

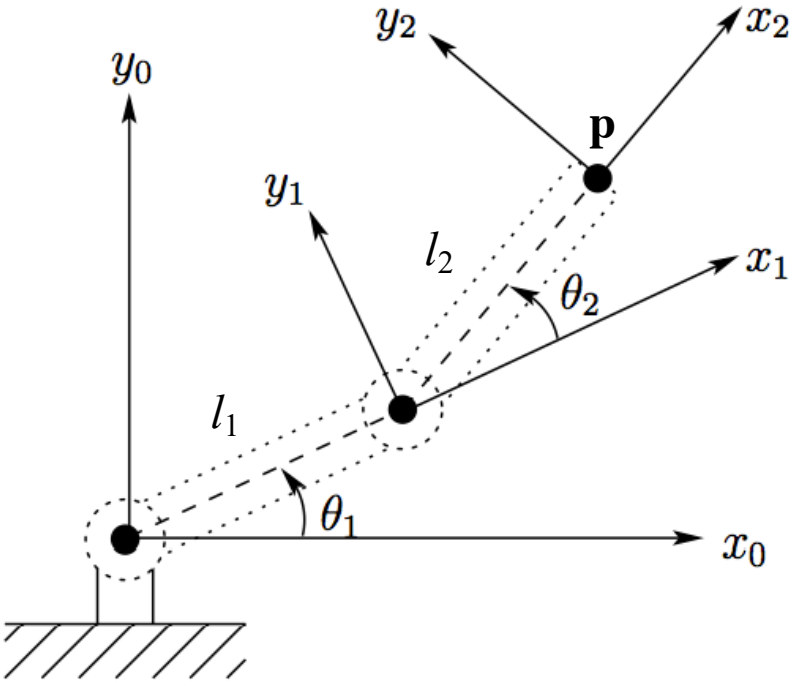
$$J_{ij} = \frac{\partial f_i}{\partial q_j} \text{ avec } f(\mathbf{q}) = (f_1(\mathbf{q}), f_2(\mathbf{q}), \ldots, f_6(\mathbf{q}))$$

Chaque colonne i transforme la **vitesse** à l'extrémité du bras en **vitesse** de l'articulation i

Bras 2D – calcul de J

Cinématique directe :

$$p_x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$p_y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$



Dérivées partielles :

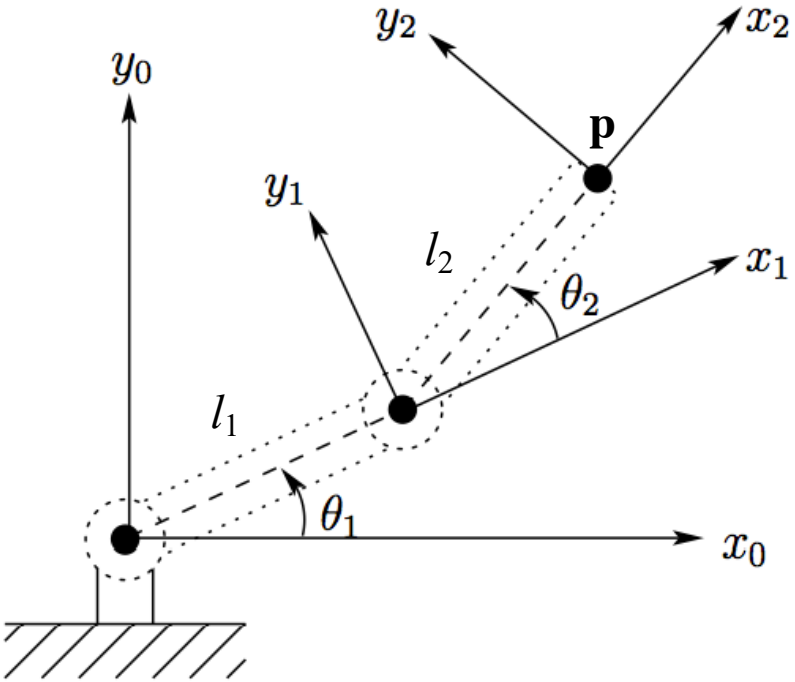
$$\frac{\partial p_x}{\partial \theta_1} = ?$$
$$\frac{\partial p_y}{\partial \theta_1} = ?$$
$$\frac{\partial p_x}{\partial \theta_2} = ?$$
$$\frac{\partial p_y}{\partial \theta_2} = ?$$

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \\ \frac{\partial p_y}{\partial \theta_1} & \frac{\partial p_y}{\partial \theta_2} \end{bmatrix}$$

Bras 2D – calcul de J

Cinématique directe :

$$p_x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$p_y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$



Dérivées partielles :

$$\frac{\partial p_x}{\partial \theta_1} = -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2)$$
$$\frac{\partial p_y}{\partial \theta_1} = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$\frac{\partial p_x}{\partial \theta_2} = -l_2 \sin(\theta_1 + \theta_2)$$
$$\frac{\partial p_y}{\partial \theta_2} = l_2 \cos(\theta_1 + \theta_2)$$

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \\ \frac{\partial p_y}{\partial \theta_1} & \frac{\partial p_y}{\partial \theta_2} \end{bmatrix}$$

Bras 2D – calcul de J

vitesse à l'extrémité du bras

$$\dot{\mathbf{p}} = J\dot{\mathbf{q}} \quad \text{avec } J = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin\theta_1 + \theta_2 \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

vitesse de la configuration

D'où : $\dot{\mathbf{q}} = J^{-1}\dot{\mathbf{p}}$

avec $J^{-1} = \dfrac{1}{l_1 l_2 \sin(\theta_2)} \begin{bmatrix} l_2 \cos(\theta_1 + \theta_2) & l_2 \sin(\theta_1 + \theta_2) \\ -l_1 \cos\theta_1 - l_2 \cos(\theta_1 + \theta_2) & -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) \end{bmatrix}$

Calcul de J

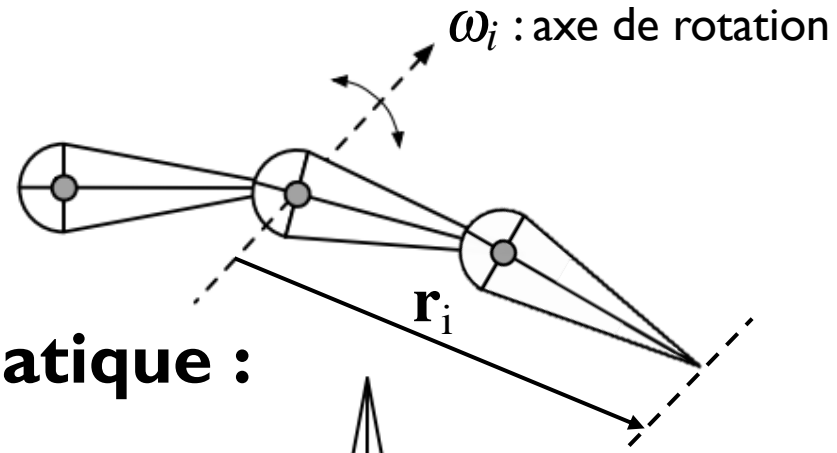
$$J = [J_1 J_2 \cdots J_n]$$

divisée en deux sous-matrices 3xn :

$$J = \begin{bmatrix} J_{\mathbf{v}} \\ J_{\omega} \end{bmatrix} \begin{array}{l} \leftarrow \text{vitesse linéaire} \\ \leftarrow \text{vitesse angulaire} \end{array}$$

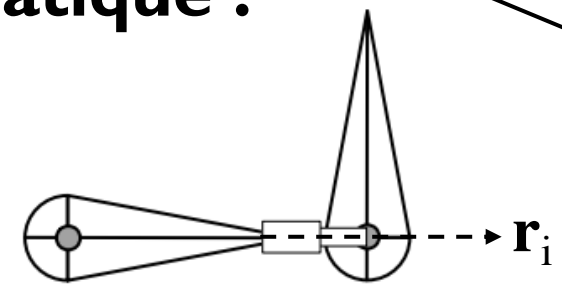
Pour une rotule :

$$J_i = \begin{bmatrix} \omega_i \times \mathbf{r}_i \\ \omega_i \end{bmatrix}$$



Pour une jonction prismatique :

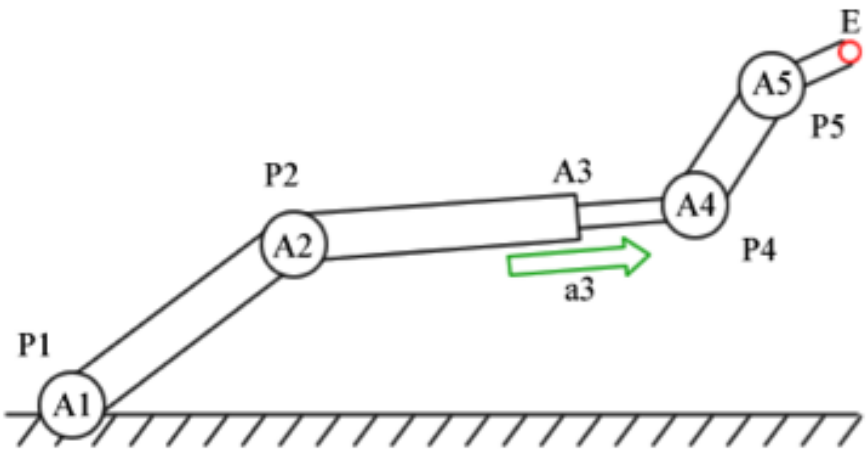
$$J_i = \begin{bmatrix} \mathbf{r}_i \\ \mathbf{0} \end{bmatrix}$$



Attention : toutes les quantités doivent être exprimées **en espace monde !**

Calcul de J – Exercice

On considère le bras articulé suivant :



1. A1, A2, A4 et A5 sont des rotules à 1 degré de liberté dont l'axe de rotation est **z**
2. A3 est une articulation prismatique d'axe **a3**
3. P1, P2, P4 et P5 sont les positions des rotules, et P1 = (0,0,0)
4. E est la position cible

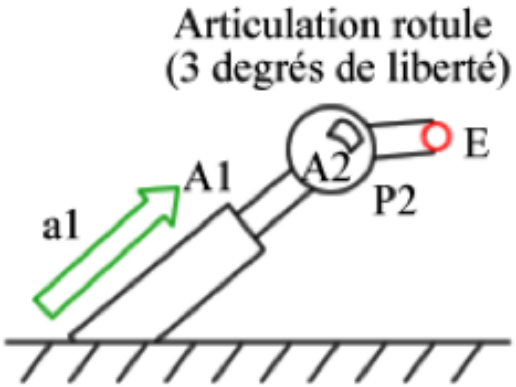
Donnez le Jacobien géométrique correspondant.

$$J = \begin{bmatrix} \overset{A1}{\mathbf{z} \times E} & \overset{A2}{\mathbf{z} \times (E - P2)} & \overset{A3}{\mathbf{a3}/||\mathbf{a3}||} & \overset{A4}{\mathbf{z} \times (E - P4)} & \overset{A5}{\mathbf{z} \times (E - P5)} \\ \mathbf{z} & \mathbf{z} & \mathbf{0} & \mathbf{z} & \mathbf{z} \end{bmatrix}$$

avec $\mathbf{z} = (0,0,1)^\top$

Calcul de J – Exercice

On considère le bras articulé suivant :



1. A1 est une articulation prismatique d'axe **a1**
2. A2 est une rotule à 3 degrés de liberté et P2 est sa position
3. E est la position cible

Remarque : une rotule à 3 degrés de liberté est équivalente à 3 rotules à 1 degré de liberté

Donnez le Jacobien géométrique correspondant.

$$J = \begin{bmatrix} \overbrace{\mathbf{a1}/||\mathbf{a1}||}^{\text{A1}} & \overbrace{\mathbf{x}\times(E-P2)}^{\text{A2}} & R(\theta_x)\mathbf{y}\times(E-P2) & R(\theta_x)R(\theta_y)\mathbf{z}\times(E-P2) \\ \mathbf{0} & \mathbf{x} & R(\theta_x)\mathbf{y} & R(\theta_x)R(\theta_y)\mathbf{z} \end{bmatrix}$$

avec $\mathbf{x} = (1,0,0)^\top$, $\mathbf{y} = (0,1,0)^\top$, $\mathbf{z} = (0,0,1)^\top$

et $R(\theta_x)$ (resp. $R(\theta_y)$) la rotation d'axe $\mathbf{x}$ et d'angle θ_x (resp. d'axe $\mathbf{y}$ et d'angle θ_y)

Algorithme

Tant que $\Delta\mathbf{p}_i > seuil$ et $\mathrm{i} < max\_iterations$

$\Delta\mathbf{p}_i = \mathbf{p} - f(\mathbf{q}_i)$ *# calcul de l'erreur*

$\Delta\mathbf{q}_i = J(\mathbf{q}_i)^{-1}\,\Delta\mathbf{p}_i$ *# calcul de la « direction » du pas*

$\mathbf{q}_{i+1} = \mathbf{q}_i + \gamma\,\Delta\mathbf{q}_i$ *# nouvelle configuration*

γ permet de contrôler la longueur du pas

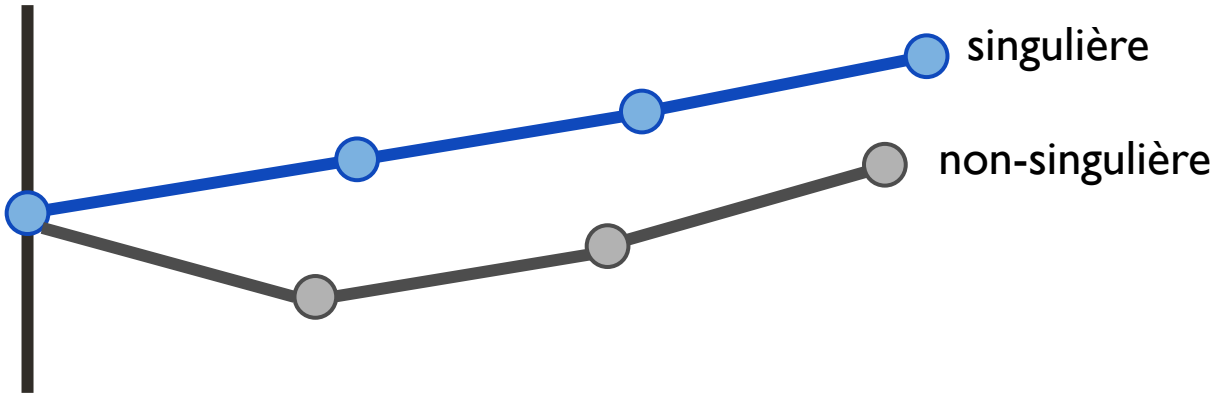
Inversion de J

Matrice rectangulaire $6 \times n \Longrightarrow$ **pseudo-inverse** J^+

$$J^+ = J^\top (JJ^\top)^{-1} \text{ si } n > 6$$
$$= (J^\top J)^{-1} J^\top \text{ si } n < 6$$

Problèmes :

- Assez lent à calculer $O(m^2 n)$
- Instable autour des singularités (os alignés)



Algorithme

Tant que $\Delta\mathbf{p}_i > seuil$ et $\mathrm{i} < max\_iterations$

$\Delta\mathbf{p}_i = \mathbf{p} - f(\mathbf{q}_i)$ *# calcul de l'erreur*

$\Delta\mathbf{q}_i = J(\mathbf{q}_i)^{+}\,\Delta\mathbf{p}_i$ *# calcul de la « direction » du pas*

$\mathbf{q}_{i+1} = \mathbf{q}_i + \gamma\,\Delta\mathbf{q}_i$ *# nouvelle configuration*

γ permet de contrôler la longueur du pas

Démo 2D en Javascript : http://kinematics.wylie.su

Alternative : transposée de J

On veut minimiser :

$$\begin{aligned} C(\Delta\mathbf{q}) &= \|J\Delta\mathbf{q} - \Delta\mathbf{p}\|^2 \\ &= (J\Delta\mathbf{q} - \Delta\mathbf{p})^\top (J\Delta\mathbf{q} - \Delta\mathbf{p}) \\ &= \Delta\mathbf{q}^\top J^\top J\Delta\mathbf{q} - 2\Delta\mathbf{q}^\top J^\top \Delta\mathbf{p} + \Delta\mathbf{p}^\top \Delta\mathbf{p} \end{aligned}$$

On prend la dérivée par rapport à $\Delta\mathbf{q}$:

$$dC/d\Delta\mathbf{q} = 2J^\top \mathrm{J}\Delta\mathbf{q} - 2J^\top \Delta\mathbf{p} + 0$$

On l'évalue au point de convergence, en $\Delta\mathbf{q} = 0$:

$$\Longrightarrow 2J^\top \Delta\mathbf{p}$$

Algorithme

Tant que $\Delta\mathbf{p}_i > seuil$ et $i < max\_iterations$

$\Delta\mathbf{p}_i = \mathbf{p} - f(\mathbf{q}_i)$ *# calcul de l'erreur*

$\Delta\mathbf{q}_i = J(\mathbf{q}_i)^T \Delta\mathbf{p}_i$ *# calcul de la « direction » du pas*

$\mathbf{q}_{i+1} = \mathbf{q}_i + \gamma \Delta\mathbf{q}_i$ *# nouvelle configuration*

γ permet de contrôler la longueur du pas

- ✓ moins coûteux (pas d'inversion de matrice)
- ✓ pas de problème de singularités
- ✗ convergence plus lente
- ✗ problème d'échelle
  (les articulations en début de chaîne font de plus grands pas)

Limites aux articulations

Pour assurer le réalisme,
respect de **contraintes morphologiques**

ex. : l'angle du coude varie entre 0 et π

Modifications de l'algorithme

- Tester si dépassement pour paramètre i
- Auquel cas, annuler le paramètre i
  (en pratique, enlever la colonne correspondante dans J)
- Recalculer J, J^+ ou $J^\top$ et $\Delta\mathbf{q}$ sans la colonne i
- Vérifier les autres contraintes

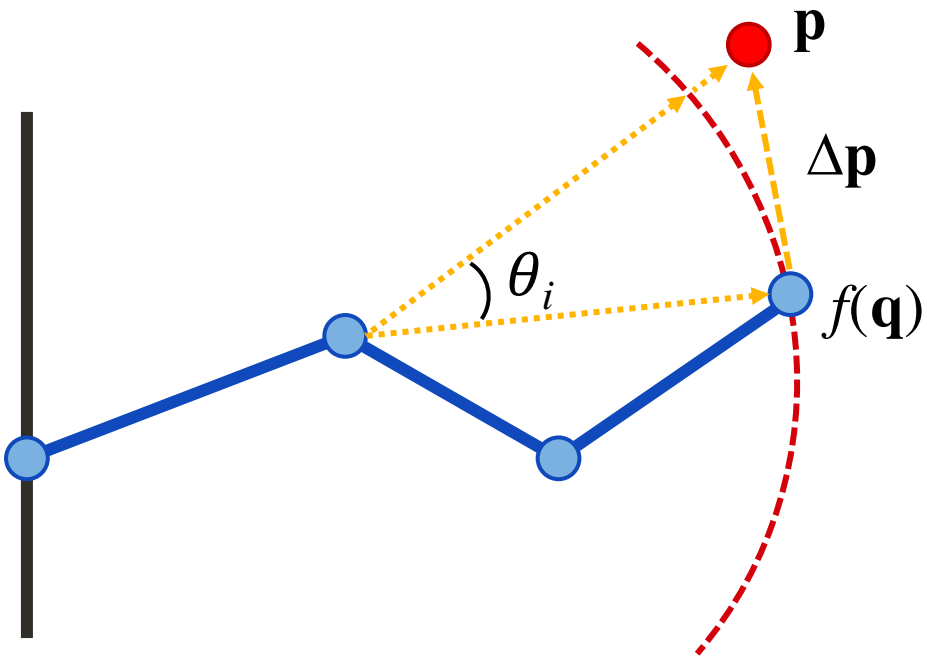
Cyclic Coordinate Descent

[Luenberger 89, Wang et al. 91]

Idée : résoudre 1 DOF à la fois

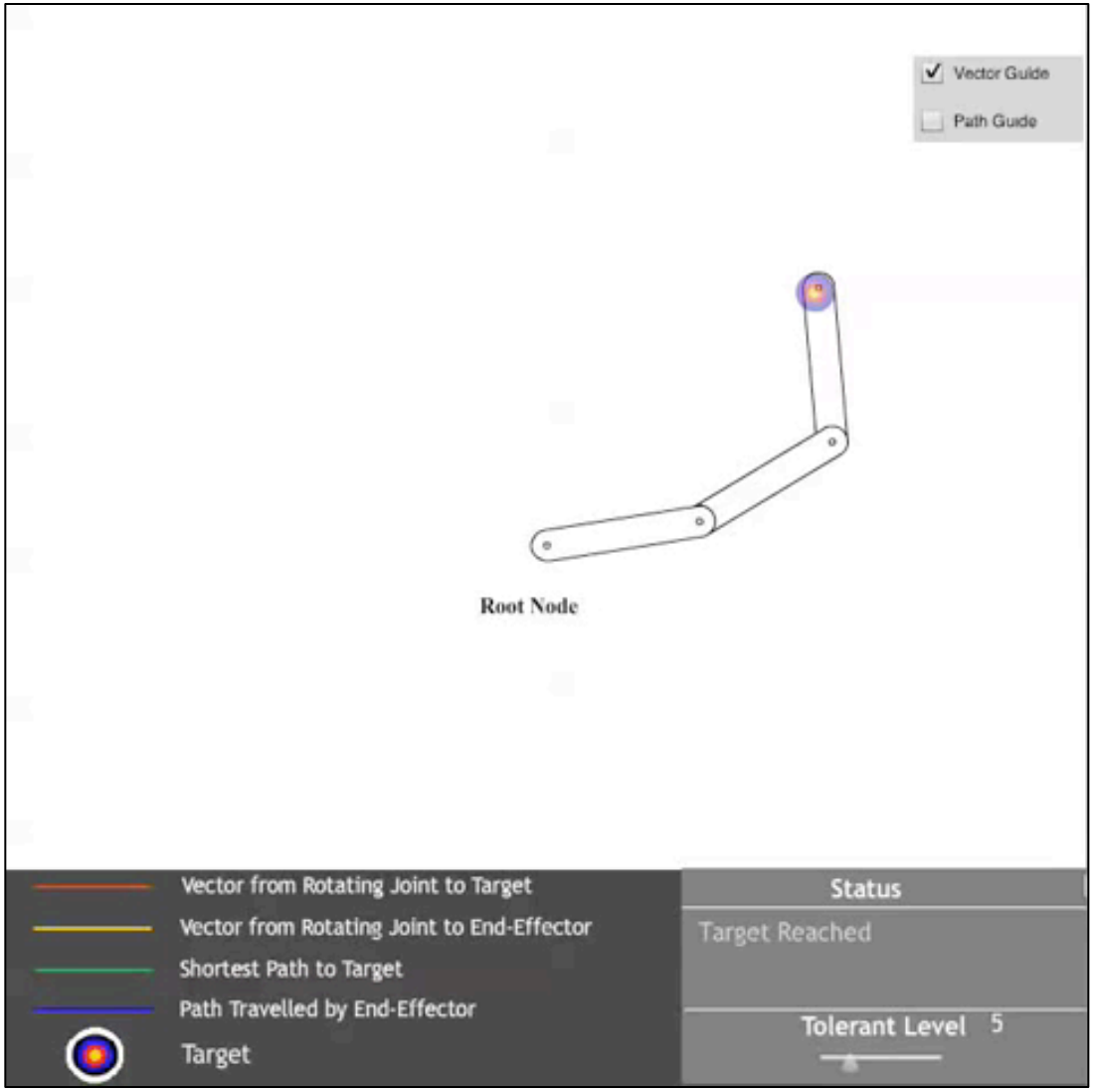
en parcourant itérativement la chaîne articulée

$\Longrightarrow$ suite de problèmes simples dont la **solution est analytique**



Trouver θ_i minimisant $\Delta\mathbf{p}$ **pour l'articulation** i

Cyclic Coordinate Descent



https://www.youtube.com/watch?v=MvuO9ZHGr6k

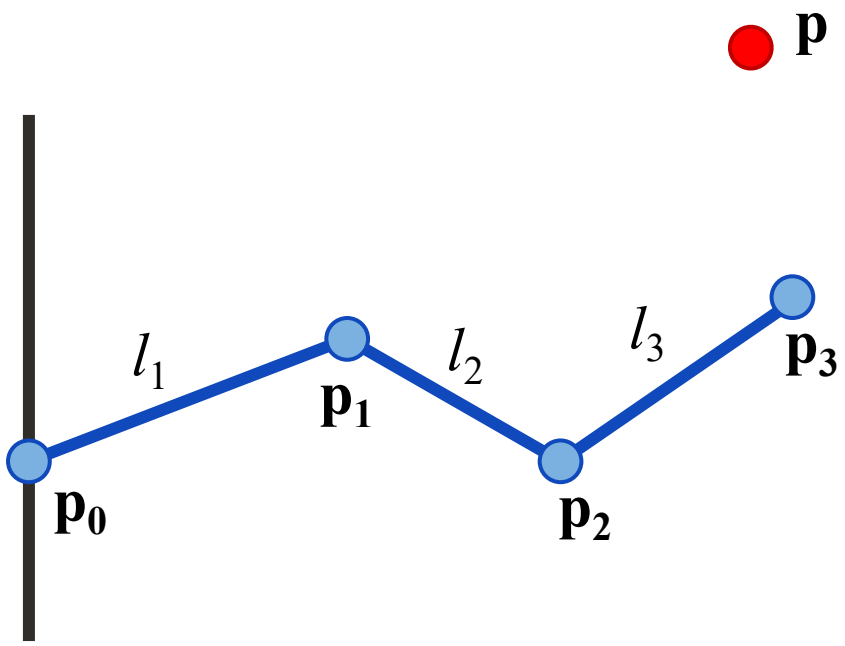
Cyclic Coordinate Descent

- ✓ simple à implémenter
- ✓ stable près des configurations singulières
- ✓ très peu coûteux (à chaque itération)
- ✗ peut nécessiter des petits pas pour converger (alors lent)
- ✗ mouvement parfois saccadé ou oscillant

Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.



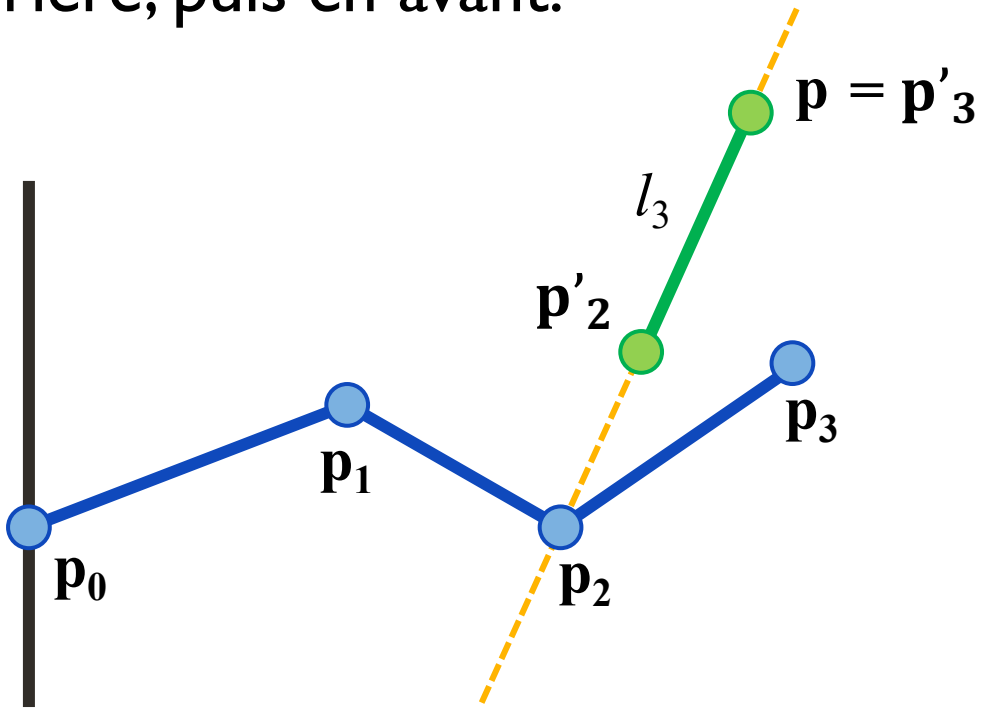
Démo 3D en Javascript : http://lo-th.github.io/fullik/

Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

1. *Forward*

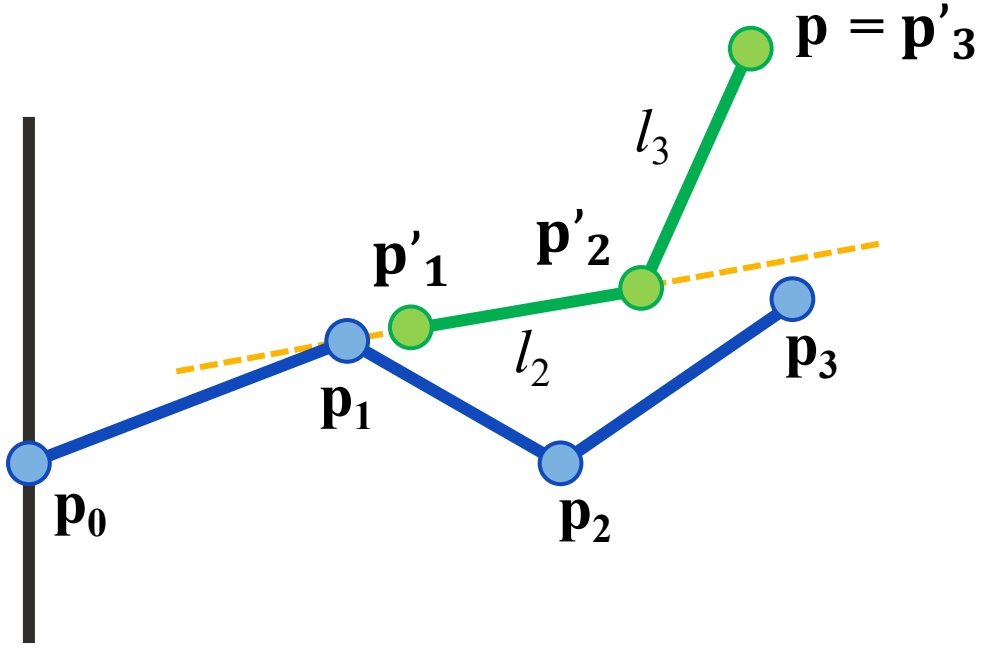


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

1. *Forward*

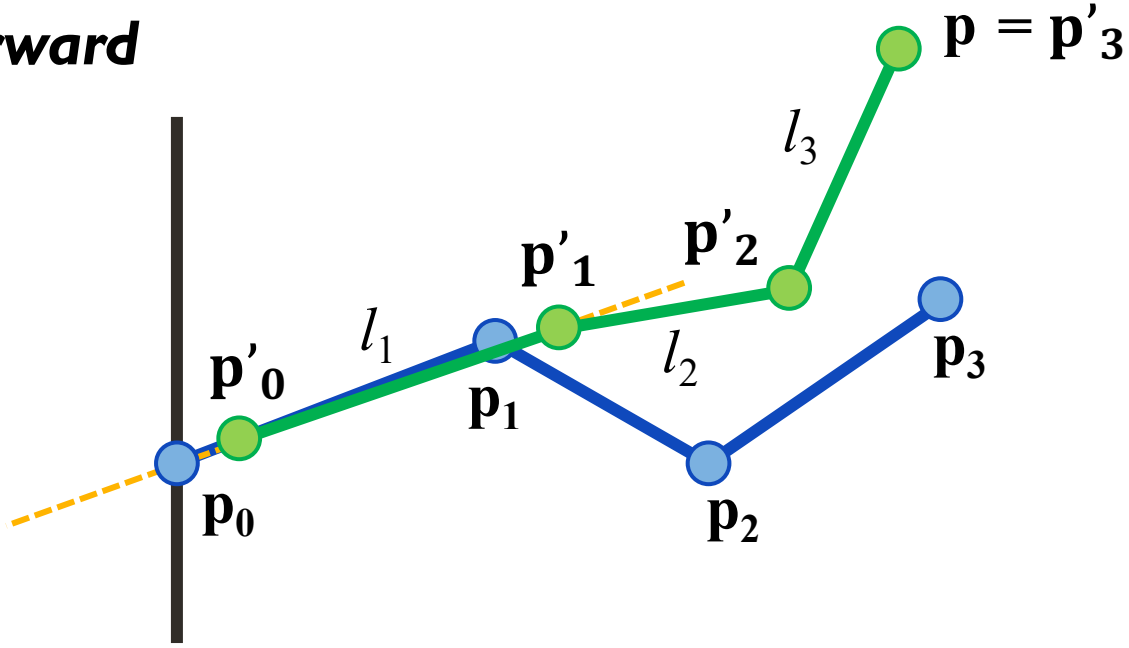


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

1. *Forward*

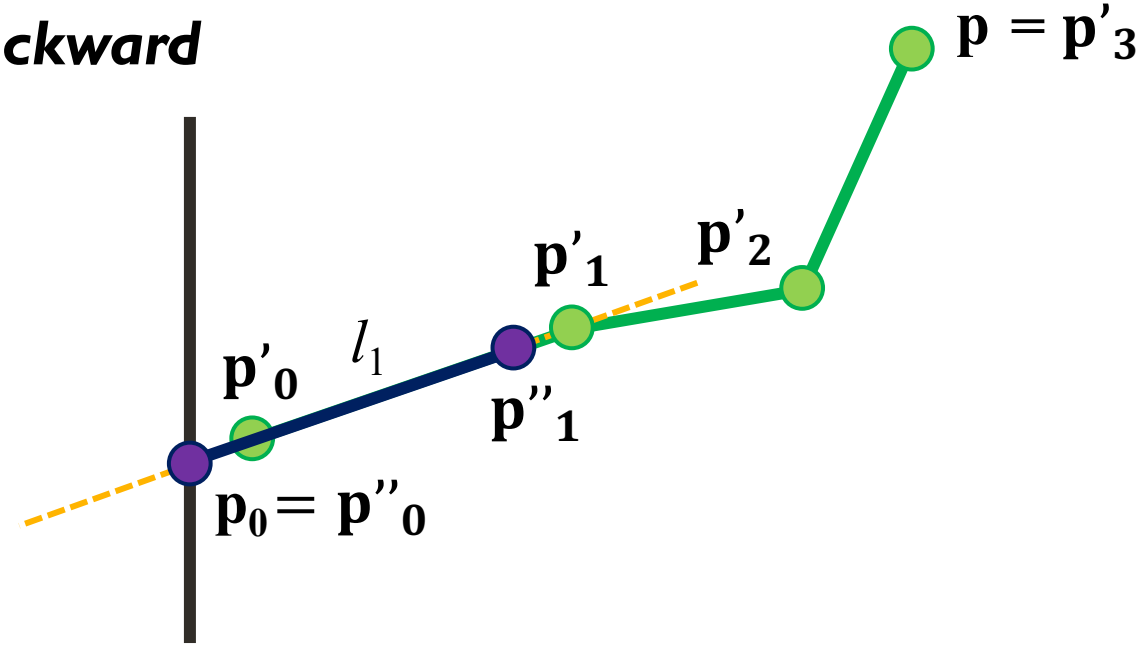


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

2. *Backward*

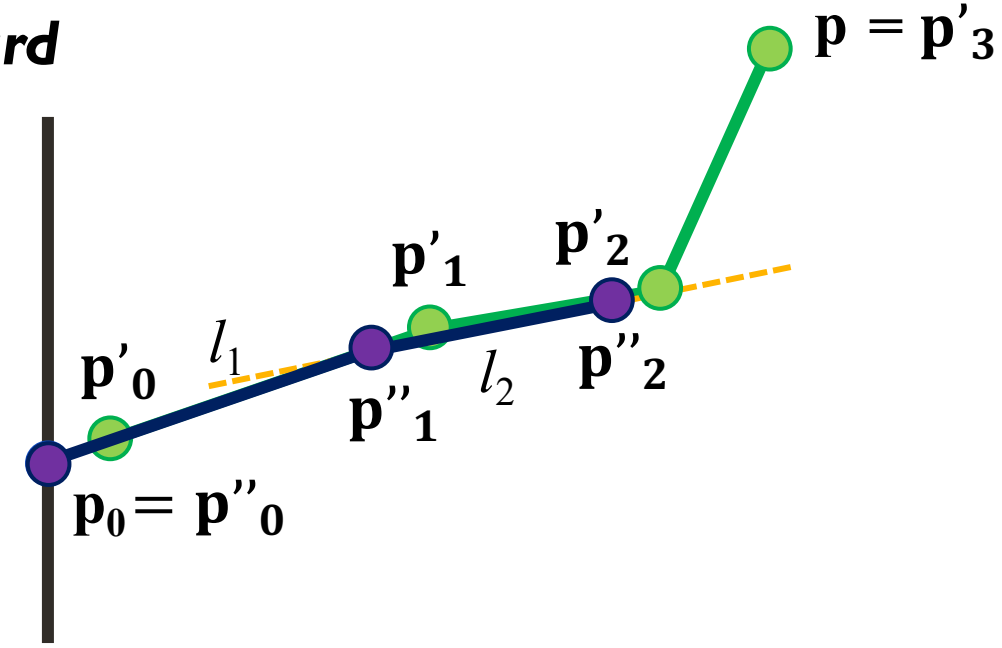


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

2. *Backward*

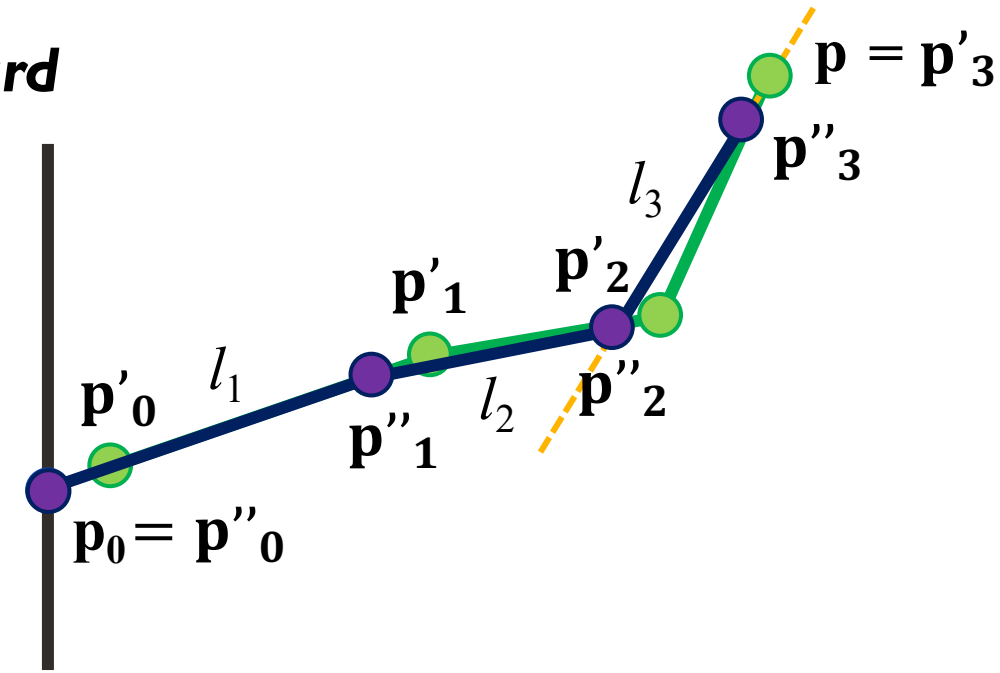


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

2. *Backward*

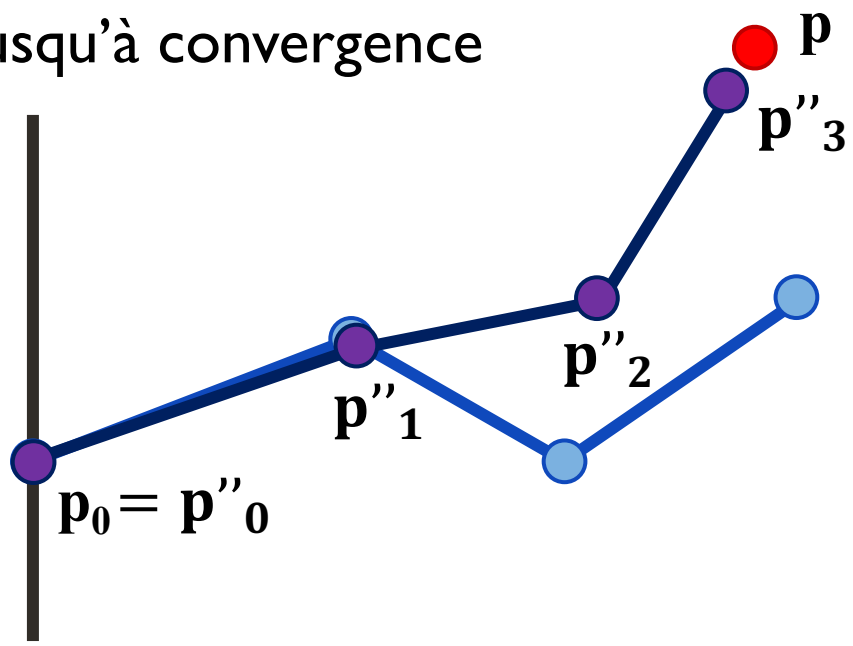


Forward And Backward Reaching IK

[Aristidou et al. 2011 & 2015]

Idée : **trouver les positions des articulations**
en parcourant itérativement la chaîne articulée
en arrière, puis en avant.

Répéter jusqu'à convergence



Forward And Backward Reaching IK

- ✓ simple à implémenter
- ✓ stable près des configurations singulières
- ✓ très, très peu coûteux à chaque itération et converge vite
- ✓ distribue le mouvement sur toute la chaîne
- ✗ résolution dans l'espace des positions
  $\Longrightarrow$ rotations à retrouver a posteriori
  $\Longrightarrow$ contraintes sur les angles peuvent entrainer des blocages (*deadlocks*)