Modèles géométriques: directs et inverses

Ludovic Hofer

6 novembre 2019

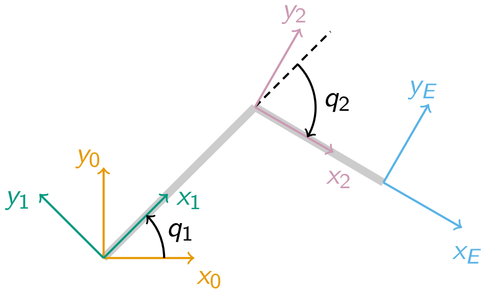
Robots articulés et modèle géométriques

- Deux types d'articulations
  - Rotoïde (liaisons angulaires)
  - Linéaire (liaisons prismatiques)
- Deux types d'architecture
  - Séries
  - Parallèles

Contenu du cours

Modèle géométrique pour robots séries

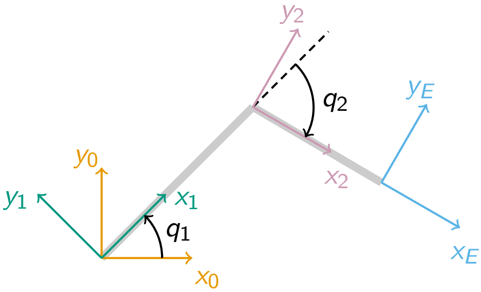
Vocabulaire



Espace articulaire : $\mathcal{Q}$

- n : nombre d'articulations (degrés de liberté)
- Configuration du robot $q = (q_1, \ldots, q_n)$
  - Ici : $q = (\pi/6, -5\pi/12)$

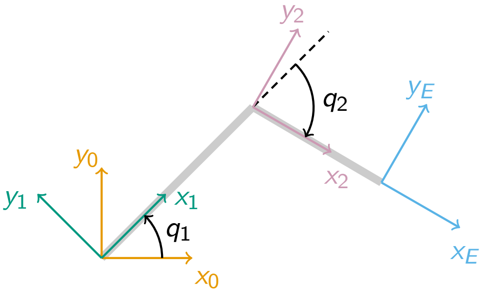
Vocabulaire



Les repères du robot

- 0 : La base du robot
- i : Repère du corps rigide après i articulations
- E : L'effecteur ou outil du robot (pince, etc...)

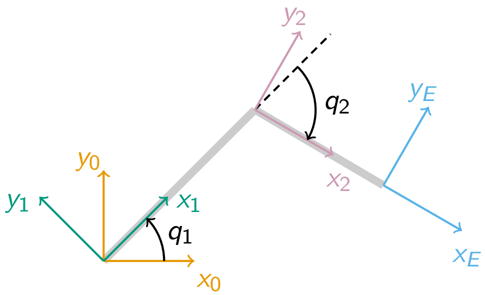
Vocabulaire



Espace opérationnel : $\mathcal{O}$

- Position de l'outil dans le repère 0
- Orientation du repère E dans le repère 0

Vocabulaire



Les modèles géométriques

Direct : $\mathcal{Q} \rightarrow \mathcal{O}$ (MGD, *Forward Kinematics*)

Inverse : $\mathcal{O} \rightarrow \mathcal{Q}$ (MGI, *Inverse Kinematics*)

Repères

Note :

On se restreint ici aux repères orthonormés directs

Notations

O_i	L'origine du repère i
$\vec{x_i}$	Le vecteur unitaire $\vec{x}$ du repère i
${}^i\vec{v}$	Le vecteur $\vec{v}$ dans le repère i
${}^iP$	La position du point P dans le repère i

Remarque

O_i, $\vec{x_i}$ et $\vec{y_i}$ suffisent à spécifier un repère, car :

$$\vec{z_i} = \vec{x_i} \wedge \vec{y_i}$$

Rotation en 3D

Notation

${}^{i}R_{j}$ La rotation de j vers i

$$ {}^{i}R_{j} = \begin{pmatrix} {}^{i}\vec{x_j} & {}^{i}\vec{y_j} & {}^{i}\vec{z_j} \end{pmatrix} $$

Propriétés

- ${}^{i}R_{j} = {}^{j}R_{i}{}^{T}$
- ${}^{i}R_{j}{}^{-1} = {}^{j}R_{i}$
- ${}^{i}R_{j}\,{}^{j}\vec{v} = {}^{i}\vec{v}$
- ${}^{i}R_{j}\,{}^{j}R_{k} = {}^{i}R_{k}$

Transformation homogène en 3D

Notation

${}^{j}T_{i}$ Transformation du repère j vers le repère i

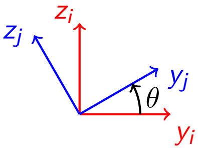
$$ {}^{i}T_{j} = \begin{pmatrix} {}^{i}R_{j} & {}^{j}\vec{O}_{i} \\ 0_{1\times 3} & 1 \end{pmatrix} $$

Propriétés

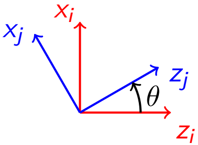
- ${}^{i}T_{j}{}^{j}T_{k} = {}^{i}T_{k}$
- ${}^{i}\vec{O}_{j} = -{}^{i}R_{j}{}^{j}\vec{O}_{i}$
- ${}^{i}T_{j}{}^{-1} = {}^{j}T_{i} = \begin{pmatrix} {}^{i}R_{j}{}^{T} & {}^{i}\vec{O}_{j} \\ 0_{1\times 3} & 1 \end{pmatrix} = \begin{pmatrix} {}^{i}R_{j}{}^{T} & -{}^{i}R_{j}{}^{T}{}^{j}\vec{O}_{i} \\ 0_{1\times 3} & 1 \end{pmatrix}$

Rotation autour des axes unitaires

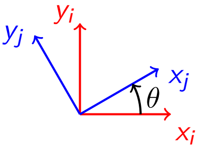
$$\mathcal{R}(\vec{x},\theta) = {}^{j}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\theta) & sin(\theta) & 0 \\ 0 & -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{i}$$



$$\mathcal{R}(\vec{y},\theta) = {}^{j}\begin{pmatrix} cos(\theta) & 0 & -sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ sin(\theta) & 0 & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{i}$$



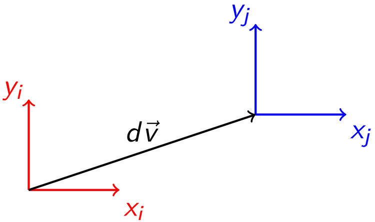
$$\mathcal{R}(\vec{z},\theta) = {}^{j}\begin{pmatrix} cos(\theta) & sin(\theta) & 0 & 0 \\ -sin(\theta) & cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{i}$$



Translation selon un axe

- $\vec{v} \in \mathbb{R}^n$: un vecteur unitaire indiquant l'axe de translation
- $d \in \mathbb{R}$: La distance de la translation

$$\mathcal{T}(\vec{v}, d) = {}^{j}\begin{pmatrix} I_{3\times 3} & -d\vec{v} \\ 0 & 1 \end{pmatrix}_{i}$$



Dérivée des transformations

- Dérivée élément par éléments

Exemple : Rotation

$$\frac{d}{dq}\overbrace{\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & cos(q) & sin(q) & 0\\ 0 & -sin(q) & cos(q) & 0\\ 0 & 0 & 0 & 1\end{pmatrix}}^{\mathcal{R}(\vec{x},q)} = \overbrace{\begin{pmatrix}0 & 0 & 0 & 0\\ 0 & -sin(q) & cos(q) & 0\\ 0 & -cos(q) & -sin(q) & 0\\ 0 & 0 & 0 & 0\end{pmatrix}}^{\mathcal{R}'(\vec{x},q)}$$

Transformation en fonction de q

- À chaque $q \in \mathcal{Q}$ correspond une matrice ${}^0T_E$
- Architecture série : transformations successives :

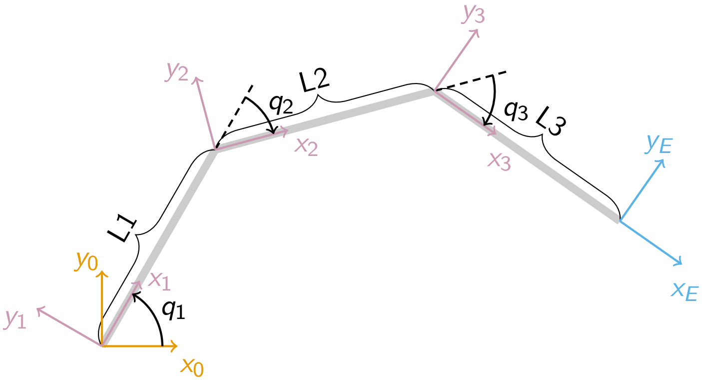
$$ {}^0T_E(q) = {}^0T_1(q_1)\,{}^1T_2(q_2)\ldots{}^{n-1}T_n(q_n)\,{}^nT_E $$

- Chaque transformation peut encore être décomposée

$$ {}^iT_j = {}^iT_{j''}\,{}^{j''}T_{j'}(q_i)\,{}^{j'}T_j $$

  - Rotoïde : ${}^{j''}T_{j'}(q_i)$: $\mathcal{R}(\vec{v}, q_i)$, avec $\vec{v}$ l'axe de rotation
  - Linéaire : ${}^{j''}T_{j'}(q_i)$: $\mathcal{T}(\vec{v}, q_i)$, avec $\vec{v}$ l'axe de translation

Exemple : Bras à 3 degrés de libertés



$$\underbrace{\overbrace{\mathcal{R}(\vec{z},-q_1)}^{{}^0T_1(q_1)}\overbrace{\mathcal{T}(\vec{x},-L1)\,\mathcal{R}(\vec{z},-q_2)}^{{}^1T_2(q_2)}\overbrace{\mathcal{T}(\vec{x},-L2)\,\mathcal{R}(\vec{z},-q_3)}^{{}^2T_3(q_3)}\overbrace{\mathcal{T}(\vec{x},-L3)}^{{}^3T_E}}_{{}^0T_E(q)}$$

Notes sur l'espace opérationnel

- ${}^0T_E$ contient des informations redondantes
  - 9 valeurs pour décrire l'orientation
  - 3 valeurs suffiraient
- ${}^0T_E$ contient parfois des informations inutiles
  - Par exemple, intérêt uniquement pour la position (x,y)

Notation

- $\mathcal{P}\left({}^0T_E\right)$: transformation $\rightarrow$ l'espace opérationnel
- $\mathcal{G}(q) = \mathcal{P}\left({}^0T_E(q)\right)$
- $\mathcal{G}_i(q)$ Le i-ème élément du vecteur $\mathcal{G}(q)$

Quelques exemples de projection dans $\mathcal{O}$

Position et orientation

$$\mathcal{P}\left({}^0T_E\right) = \mathcal{P}\left(\begin{pmatrix} {}^0R_E & {}^0O_E \\ 0_{1\times 3} & 1 \end{pmatrix}\right) = \begin{pmatrix} {}^0O_E \\ \left.\begin{matrix} \text{roll} \\ \text{pitch} \\ \text{yaw} \end{matrix}\right\} {}^0R_E \end{pmatrix}$$

Position uniquement

$$\mathcal{P}\left({}^0T_E\right) = \mathcal{P}\left(\begin{pmatrix} {}^0R_E & {}^0O_E \\ 0_{1\times 3} & 1 \end{pmatrix}\right) = {}^0O_E$$

Modèle Géométrique Inverse

Objectif

Pour un $o \in \mathcal{O}$, quelles configurations $q \in \mathcal{Q}$ tel que $\mathcal{G}(q) = o$

Différences avec le MGD

- Généralement, plusieurs solutions
- Parfois 0 solutions
- Parfois infinité de solutions

MGI et nombre de degrés de liberté

Cas classique : $n = 6$

L'espace opérationnel comprend position et orientation.

Sur-contraint : $n < 6$

Suppression de contraintes (par exemple position uniquement)

Sous-contraint : $n > 6$

Plusieurs possibilités :

- Fixer toutes les articulations sauf 6
- Introduire des contraintes supplémentaires

Méthodes de résolutions

Pas de solution générale, mais deux approches du problème :

- Méthodes analytiques
  - Résolution géométrique
  - Résolution algébrique
- Méthodes numériques (itératives)
  - Par Jacobienne Inverse
  - Par Jacobienne Transposée

Avantages et inconvénients

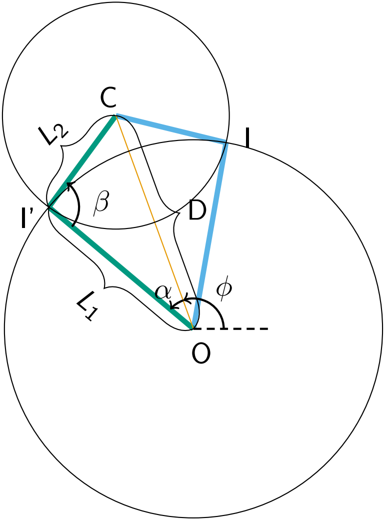
Avantages

- Réponses exactes
- Nombre de solutions disponible
- Exécution rapide

Inconvénients

- Pas de méthode générale : propre à chaque robot
- Ne fournit pas de solution approchée quand la cible n'est pas atteignable

Résolution géométrique : cas pratique



Données du problèmes

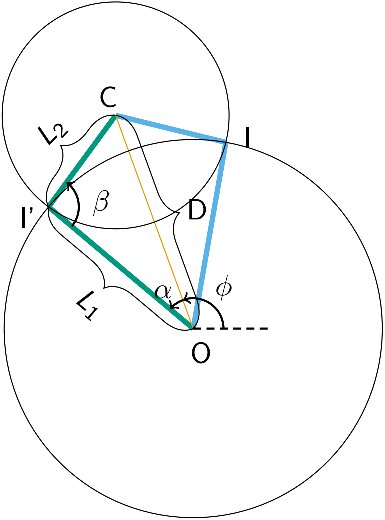
C : Position cible

D : $\|O - C\|$

L_1 : $\|O - I\|$

L_2 : $\|I - C\|$

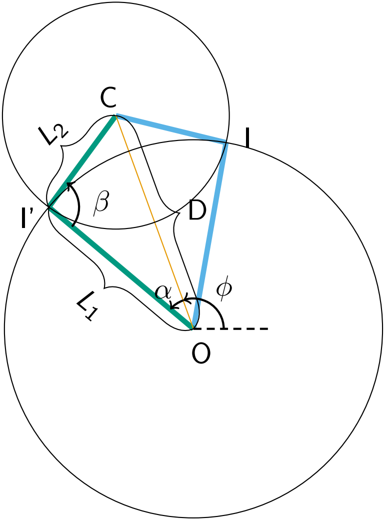
Résolution géométrique : cas pratique



Quelles valeurs pour α et β

- Par *Al-Kashi* :
  $${L_2}^2 = D^2 + {L_1}^2 - 2DL_2\cos(\alpha)$$
- Autrement dit :
  $$\alpha = \arccos\left(\frac{{L_1}^2+D^2-{L_2}^2}{2L_1D}\right)$$
- De manière similaire :
  $$\beta = \arccos\left(\frac{{L_1}^2+{L_2}^2-D^2}{2L_1L_2}\right)$$

Résolution géométrique : cas pratique

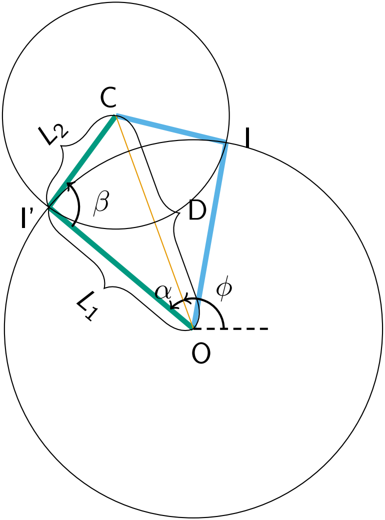


Quelles valeurs pour q_1 et q_2

- Cas classique : 2 solutions :

$$\begin{cases} q_1 = \phi - \alpha, q_2 = 180 - \beta \\ q_1 = \phi + \alpha, q_2 = \beta - 180 \end{cases}$$

Résolution géométrique : cas pratique



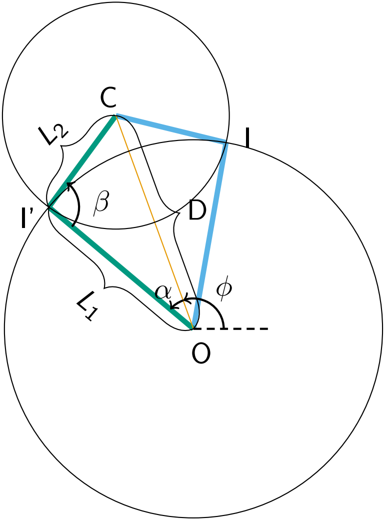
Quelles valeurs pour q_1 et q_2

- Cas classique : 2 solutions :

$$\begin{cases} q_1 = \phi - \alpha, q_2 = 180 - \beta \\ q_1 = \phi + \alpha, q_2 = \beta - 180 \end{cases}$$

- Pas de solutions :
  - $D > L_1 + L_2$ ou
  - $D < |L_2 - L_1|$

Résolution géométrique : cas pratique



Quelles valeurs pour q_1 et q_2

- Cas classique : 2 solutions :

$$\begin{cases} q_1 = \phi - \alpha, q_2 = 180 - \beta \\ q_1 = \phi + \alpha, q_2 = \beta - 180 \end{cases}$$

- Pas de solutions :
  - $D > L_1 + L_2$ ou
  - $D < |L_2 - L_1|$
- Une seule solution :
  $D = L_1 + L_2$

Résolution algébrique

- Non-couvert ici [1]
- Systèmes d'équation avec cos et sin
- Choix du repère dans lequel sont exprimés est important
- Utilisation de calcul symbolique (sympy, maxima, maple)

1. Voir 1.2.3 : `http://cours-online.gdr-robotique.org/Khalil-Dombre_Modelisation/Khalil-Dombre_Modelisation.pdf`

La jacobienne

$$J(q) = \begin{pmatrix} \frac{\partial \mathcal{G}(q)}{\partial q_1} & \dots & \frac{\partial \mathcal{G}(q)}{\partial q_n} \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathcal{G}_1(q)}{\partial q_1} & \dots & \frac{\partial \mathcal{G}_1(q)}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathcal{G}_k(q)}{\partial q_1} & \dots & \frac{\partial \mathcal{G}_k(q)}{\partial q_n} \end{pmatrix}$$

Utilités

- MGI : méthodes numériques
- Vitesse dans l'espace opérationnel : $\dot{o} = J(q)\dot{q}$

Avantages et inconvénients

Avantages

- Méthode similaire pour tous les robots, basée sur $\mathcal{G}$
- Solution approximative pour position impossible

Inconvénients

- Fournit une seule solution
- Vulnérable aux singularités
- Calculatoire
- Non répétable

Inverse de la jacobienne

Résolution du MGI

Soit $\epsilon \in \mathbb{R}^k$ un vecteur de norme faible :
On linéarise $\mathcal{G}$ autour de q : $\mathcal{G}(q+\epsilon) \approx \mathcal{G}(q) + J(q)\epsilon$
Donc : $o - \mathcal{G}(q) \approx J(q)\epsilon$
On peut donc trouver ϵ avec : $\epsilon \approx J(q)^{-1}(o - \mathcal{G}(q))$

Problèmes fréquents

- ϵ trop grand : approximation linéaire invalide
    - Plusieurs pas de résolution : méthode itérative
- $J(q)$ non-inversible (exemple : matrice rectangulaire)

Jacobienne Transposée : Théorie

Formulation du problème

Minimisation d'une fonction de coût $C(o, q)$ avec :

- $o \in \mathcal{O}$: la cible à atteindre
- $q \in \mathcal{Q}$: la configuration du robot

Recherche du coût minimum

- Optimisation de fonction en boîte noire
- Résolution plus efficace avec accès au gradient : $\nabla C(o, q)$

$$\nabla C(o, q) = \begin{pmatrix} \frac{\partial}{\partial q_1} C(o, q) \\ \vdots \\ \frac{\partial}{\partial q_n} C(o, q) \end{pmatrix}$$

Jacobienne Transposée : Exemple

Cas simple

- Cas simple : 3 degrés de liberté
- Cible : position en 3D
- Coût : carré des erreurs :

$$C(o,q) = \sum_{i=0}^{3} (o_i - \mathcal{G}_i(q))^2 = (o - \mathcal{G}(q))^T (o - \mathcal{G}(q))$$

Jacobienne Transposée : calcul de $\nabla C(o, q)$

Fonction de coût : carré des erreurs :

$$C(o, q) = \sum_{i=0}^{3} (o_i - \mathcal{G}_i(q))^2$$

Jacobienne Transposée : calcul de $\nabla C(o, q)$

Fonction de coût : carré des erreurs :

$$C(o, q) = \sum_{i=0}^{3} (o_i - \mathcal{G}_i(q))^2$$

Dérivations de fonctions composées :

$$\frac{\partial C(o, q)}{\partial q_j} = \sum_{i=1}^{3} -2(o_i - \mathcal{G}_i(q)) \frac{\partial \mathcal{G}_i(q)}{\partial q_j}$$

Jacobienne Transposée : calcul de $\nabla C(o, q)$

Dérivations de fonctions composées :

$$\frac{\partial C(o,q)}{\partial q_j} = \sum_{i=1}^{3} -2(o_i - \mathcal{G}_i(q)) \frac{\partial \mathcal{G}_i(q)}{\partial q_j}$$

Autrement dit :

$$\frac{\partial C(o,q)}{\partial q_j} = -2 \left(\frac{\partial \mathcal{G}(q)}{\partial q_j} \right)^T (o - \mathcal{G}(q))$$

Jacobienne Transposée : calcul de $\nabla C(o, q)$

Autrement dit :

$$\frac{\partial C(o,q)}{\partial q_j} = -2\left(\frac{\partial \mathcal{G}(q)}{\partial q_j}\right)^T (o - \mathcal{G}(q))$$

D'où :

$$\nabla C(o,q) = -2\begin{pmatrix} \frac{\partial \mathcal{G}_1(q)}{\partial q_1} & \frac{\partial \mathcal{G}_2(q)}{\partial q_1} & \frac{\partial \mathcal{G}_3(q)}{\partial q_1} \\ \frac{\partial \mathcal{G}_1(q)}{\partial q_2} & \frac{\partial \mathcal{G}_2(q)}{\partial q_2} & \frac{\partial \mathcal{G}_3(q)}{\partial q_2} \\ \frac{\partial \mathcal{G}_1(q)}{\partial q_3} & \frac{\partial \mathcal{G}_2(q)}{\partial q_3} & \frac{\partial \mathcal{G}_3(q)}{\partial q_3} \end{pmatrix} (o - \mathcal{G}(q))$$

Jacobienne Transposée : calcul de $\nabla C(o,q)$

D'où :

$$\nabla C(o,q) = -2 \begin{pmatrix} \frac{\partial \mathcal{G}_1(q)}{\partial q_1} & \frac{\partial \mathcal{G}_2(q)}{\partial q_1} & \frac{\partial \mathcal{G}_3(q)}{\partial q_1} \\ \frac{\partial \mathcal{G}_1(q)}{\partial q_2} & \frac{\partial \mathcal{G}_2(q)}{\partial q_2} & \frac{\partial \mathcal{G}_3(q)}{\partial q_2} \\ \frac{\partial \mathcal{G}_1(q)}{\partial q_3} & \frac{\partial \mathcal{G}_2(q)}{\partial q_3} & \frac{\partial \mathcal{G}_3(q)}{\partial q_3} \end{pmatrix} (o - \mathcal{G}(q))$$

Finalement :

$$\nabla C(o,q) = -2J(q)^T (o - \mathcal{G}(q))$$

Pour aller plus loin

- https://www.pobot.org/IMG/pdf/cinematique_des_robots_series.pdf
- http://cours-online.gdr-robotique.org/Khalil-Dombre_Modelisation/Khalil-Dombre_Modelisation.pdf