CINEMATIQUE INVERSE APPLIQUEE AU BRAS DE ROBOT POPPY-HUMANOÏDE

Par

Xavier MARIOT

Étude entrant dans le cadre d'un stage effectué
entre la 2ème et 3ème année du cursus :

FITE de l'ENSAM
(formation ingénieur généraliste)

Arts Et Métiers ParisTech – CER Bordeaux-Talence

Été 2015

Tuteur de stage : Jean-Luc CHARLES

En partenariat avec : l'INRIA - Bordeaux

TABLE DES MATIERES

INTRODUCTION

Cette étude vise à résoudre le problème de cinématique inverse dans le pilotage des mouvements d'un robot qui est un problème couramment rencontré dans le domaine de la robotique. Le problème revient à déterminer les valeurs des paramètres (angles ou translations de servomoteurs, etc.) à imposer pour un modèle cinématique donné du robot afin que le système ou une partie du système soit dans une position voulue.

La difficulté de la résolution de ce problème est en général la non unicité de la solution permettant de donner cette position au système ou à une partie du système c.à.d. qu'il y a une infinité de valeurs possibles pour les paramètres de commande qui vont toutes amener le robot à la position voulue. Il arrive également que aucune solution n'existe car on impose trop de contraintes sur la pose que doit prendre le système, ce problème n'est pas traité dans cette étude mais se résout de manière similaire par la méthode des moindres carrés.

Ce document présente dans un premier temps un découpage des différentes problématiques et une stratégie globale de résolution de ce problème de cinématique inverse. Est ensuite exposé l'application de cette méthode au bras de robot Poppy avec les différents algorithmes et outils mathématiques utilisés permettant d'implémenter numériquement cette stratégie.

Pour donner un aspect pratique, la résolution du problème est appliquée au bras de robot humanoïde Poppy tout au long de l'étude mais peut aisément être adapté à tout autre système (jambe, buste, bras articulé 6 axes, etc.).

GLOSSAIRE ET NOTATIONS

Vecteurs et fonctions vectorielles : Tout au long de ce document, les vecteurs et les fonctions vectorielles seront notés en gras. Ex. « la fonction **f** », ou encore « l'ensemble des paramètres **q** = (q_1, …, q_n)»

Df représente la différentielle de **f**.

Le lecteur ne se laissera pas perturber par le fait que pour une application linéaire, notée ϕ par exemple, on parle indifféremment de l'application linéaire ou de la matrice associée sans nécessairement changer de notation.

Pour un vecteur **u** = (u_1, …, u_n) et un repère orthonormé R0 = ($\mathbf{x_1}$, …, $\mathbf{x_n}$), on note $M_{R0}^{\boldsymbol{u}} \in \mathbb{R}_n$ la matrice représentative des coordonnées de **u** dans R0.

Pour deux repères orthonormés R1 et R0, on note M_{R0}^{R1} la matrice de R1 dans R0 ou matrice de changement de base telle que pour tout vecteur **u** :

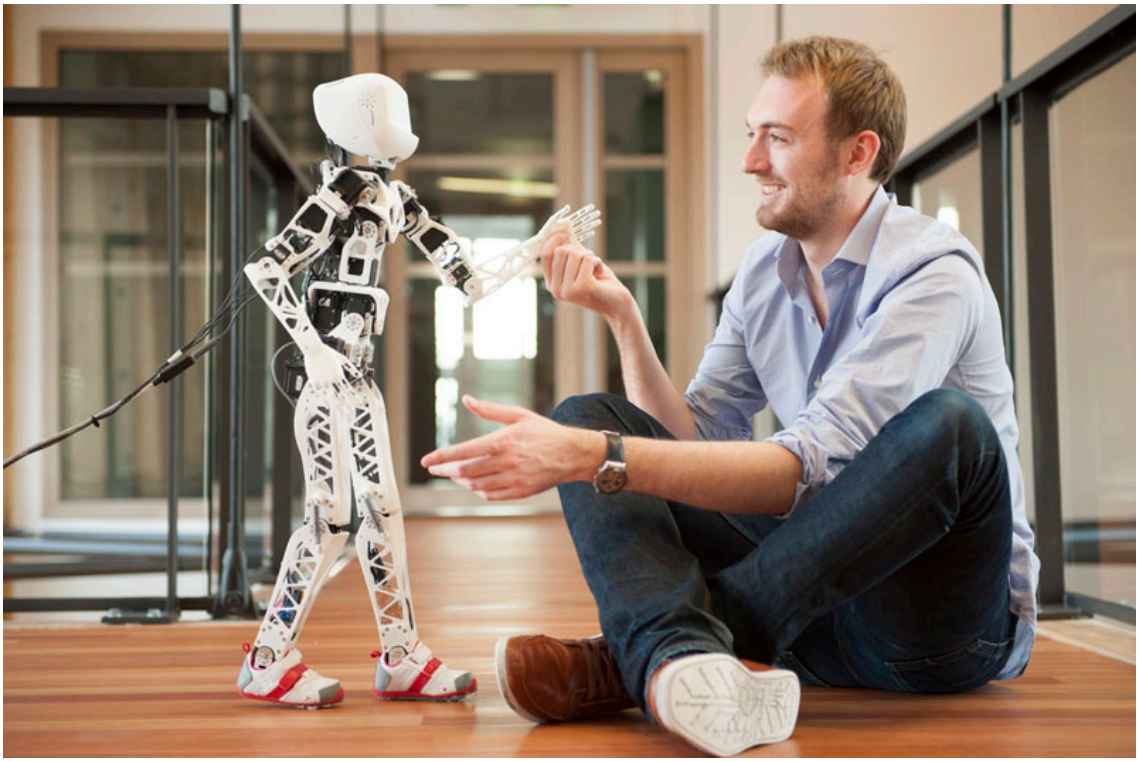
$$M_{R0}^{\boldsymbol{u}} = M_{R0}^{R1} * M_{R1}^{\boldsymbol{u}}$$

C h a p i t r e 1

PRESENTATION DU ROBOT POPPY-HUMANOÏDE

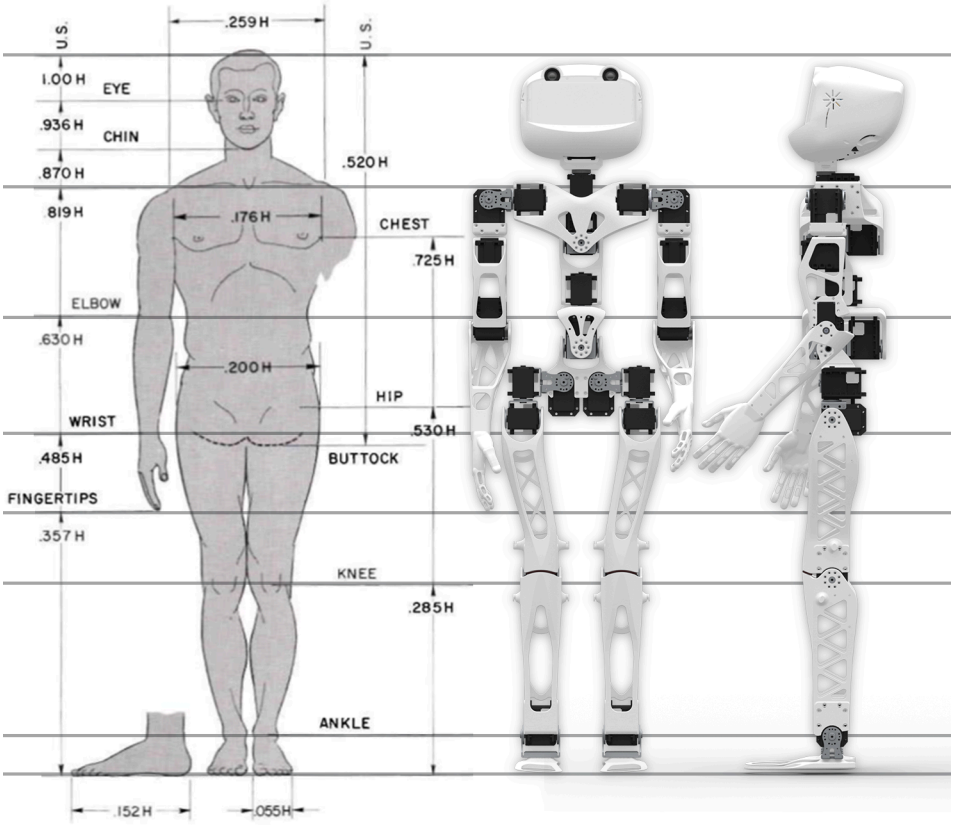
Poppy-humanoïde

Comme indiqué dans l'introduction, cette étude est appliquée au bras du robot Poppy-Humanoïde développé par le laboratoire de recherche de l'INRIA-Bordeaux.

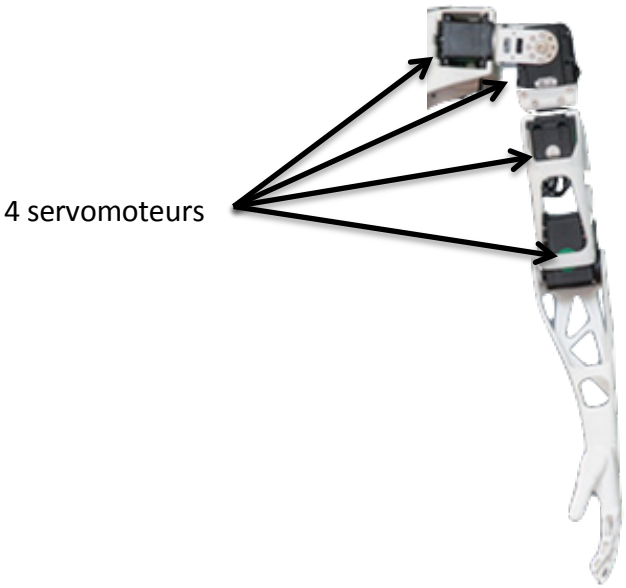


Poppy et M. LAPEYRE membre de l'équipe FLOWERS au laboratoire de l'INRIA

Son profil et ses proportions sont présentés sur le schéma suivant :



Pour mettre en place notre algorithme, nous l'appliquons au bras de ce robot humanoïde. Cela permet dans un cas simple (bras à 4 degrés de liberté) de vérifier concrètement le fonctionnement de l'algorithme en cherchant par exemple à positionner la main à une position M(x, y, z) dans le repère du buste.



Paramétrage du bras (en cours de rédaction)

Afin que le modèle qui est mis en place soit compatible avec les différents développements déjà réalisés sur Poppy-humanoïde, nous allons utiliser les différents repères représentés sur le schéma cinématique suivant. Ainsi les « zéros » des différents moteurs et les sens positifs de rotation seront respectés par rapport à la configuration « habituelle » de Poppy.

(En cours de rédaction)

ANALYSE DES PROBLEMATIQUES

Dans cette étude de cinématique inverse, il est indispensable d'identifier et de séparer les différentes problématiques que l'on cherche à résoudre. Nous pouvons distinguer trois parties majeures indépendantes.

Le premier problème est de choisir une modélisation cinématique adéquate de notre système par rapport aux objectifs de contrôle de la position. Le second problème est d'obtenir à partir de la modélisation choisie la relation de cinématique inverse. Enfin le dernier problème est de piloter le système de la position actuelle où il se trouve à la position finale obtenue par l'algorithme de cinématique inverse.

Choisir le modèle cinématique directe du système

La problématique ici est la mise en équation de la structure du robot et le choix du modèle mathématique qui représente la cinématique directe du système. En effet selon le pilotage que l'on veut pouvoir effectuer (position (x, y, z) dans un repère, positions angulaires, etc.) certains modèles mathématiques sont plus adaptés que d'autres.

Prenons le cas du bras de robot Poppy-humanoïde. Dans un cas on veut piloter la position de la main de Poppy, par exemple le bout de l'index représenté par un point M de coordonnées (x, y, z) dans le repère du buste. Dans ce cas un modèle vectoriel simple c.à.d. un vecteur **OM** qui représente la position de l'index en fonction des quatre angles des servomoteurs et des dimensions du bras suffit comme modèle cinématique directe. Par contre dans un autre cas où l'on voudrait par exemple faire garder une position horizontale à l'avant bras de Poppy et déplacer sa main, ce modèle n'est plus adapté car il ne rend pas compte de la position angulaire du repère du bras par rapport au repère du buste.

A l'issue de cette étape nous devrons avoir un modèle mathématique exploitable par l'algorithme de cinématique inverse qui suit.

Trouver les paramètres de la position finale

Il s'agit ici de trouver l'algorithme de cinématique inverse qui va retourner les valeurs des paramètres pilotables directement sur le système pour que le robot soit dans une position voulue. Attention il ne s'agit pas ici de piloter le robot, ce n'est pas le rôle de cet algorithme ! Il doit simplement répondre à la question : « Si je veux que mon système soit dans telle position, quelles sont les valeurs qu'auraient les différents paramètres d'entrée de mon système ? ».

Reprenons le cas de l'index du bras de Poppy, la question posée est : Quelles sont les valeurs des quatre angles des servomoteurs pour que l'index soit à la position (x_1, y_1, z_1) à partir du modèle cinématique direct mis en place ? L'algorithme doit renvoyer les quatre valeurs qu'ont les servomoteurs lorsque l'index est à cette position.

La difficulté majeure est qu'il peut y avoir aucune, une seule ou une infinité de solutions. Comme indiqué dans l'introduction, nous traiterons dans cette étude uniquement le cas où au moins une solution existe c.à.d. une seule ou une infinité.

Piloter le système de sa position réelle initiale à la position finale

Le dernier problème à résoudre est maintenant de déplacer physiquement le système de la position où il se trouve actuellement vers la position définie à l'issue de l'algorithme précédent. Les questions qui se posent sont de savoir comment nous faisons varier les différents paramètres du système pour arriver aux paramètres finaux. Un problème qui peut se poser est le passage du système par des positions interdites, par exemple le bras de Poppy qui traverserait son corps ce qui est techniquement impossible.

STRATEGIE GLOBALE DE RESOLUTION

Les problématiques présentées précédemment induisent naturellement la stratégie de résolution qui suit.

Modélisation préliminaire de l'architecture cinématique du système

Des modèles optimisés existent, comme les paramètres de Denavit-Hartenberg, et permettent d'incorporer un maximum d'informations sur la structure et la position des repères de chaque élément du système en un minimum de paramètres. Pour des raisons de temps, ceux-ci n'ont pas été implémentés dans cette étude. Nous utilisons un modèle vectoriel simple mais tout autre modèle peut être utilisé avec les algorithmes qui suivent.

Dans le choix du modèle cinématique direct on définit donc une fonction **f** comme suit caractéristique de la cinématique du robot :

$$\boldsymbol{f} : \begin{cases} \mathbb{R}^n \rightarrow \mathbb{R}^m \\ \boldsymbol{q} = (q_1, \dots, q_n) \rightarrow \boldsymbol{x} = (x_1, \dots, x_m) = \boldsymbol{f}(\boldsymbol{q}) \end{cases}$$

Où **q** représente l'ensemble des paramètres pilotés directement sur le robot, dans le cas du bras de Poppy les quatre angles des servomoteurs

x représente l'ensemble des paramètres que l'on veut imposer sur l'état du système, par exemple les coordonnées (x, y, z) de la main de Poppy dans le repère du buste.

Obtention des paramètres inverses

Dans cet algorithme on veut obtenir la fonction inverse de **f** :

$$\boldsymbol{f}^{-1} : \begin{cases} \mathbb{R}^m \to \mathbb{R}^n \\ \boldsymbol{x} = (x_1, \dots, x_m) \to \boldsymbol{q} = (q_1, \dots, q_n) = \boldsymbol{f}^{-1}(\boldsymbol{x}) \end{cases}$$

Cependant comme mentionné précédemment, cette fonction n'existe pas dans la majorité des cas car il y a une infinité de solutions. Une première approche consiste à vouloir imposer des conditions supplémentaires sur la position du système afin qu'il ne reste qu'une unique solution. Cependant cette méthode n'est pas assez générale, elle induit une gestion « cas par cas » trop contraignante.

La méthode que nous utilisons consiste à définir une position initiale théorique du système $\mathbf{q}_{\mathbf{init}}$ qui va permettre de définir une variation minimale du vecteur **q** entre cette position initiale et la position finale $\mathbf{q}_{\mathbf{final}}$ que l'on recherche telle que $\mathbf{f}(\mathbf{q}_{\mathbf{final}}) = \mathbf{x}_{\mathbf{final}}$. En effet il existe une seule solution dans la grande majorité des cas qui minimise la norme :

$$||\mathbf{q}_{\mathbf{final}} - \mathbf{q}_{\mathbf{init}}||$$

Attention, il faut bien garder en tête que $\mathbf{q}_{\mathbf{init}}$ n'a rien à voir avec la position réelle du système, c'est une position imaginaire utilisée pour obtenir numériquement dans un algorithme une solution $\mathbf{q}_{\mathbf{final}}$ qui vérifie $\mathbf{f}(\mathbf{q}_{\mathbf{final}}) = \mathbf{x}_{\mathbf{final}}$.

On note $\Delta\mathbf{q} = \mathbf{q}_{\mathbf{final}} - \mathbf{q}_{\mathbf{init}}$ et $\Delta x = \mathbf{x}_{\mathbf{final}} - \mathbf{x}_{\mathbf{init}}$

Où $\mathbf{f}(\mathbf{q}_{\mathbf{final}}) = \mathbf{x}_{\mathbf{final}}$ et $\mathbf{f}(\mathbf{q}_{\mathbf{init}}) = \mathbf{x}_{\mathbf{init}}$

La seule inconnue est $\mathbf{q}_{\mathbf{final}}$, tous les autres éléments sont connus (NB : $\mathbf{x}_{\mathbf{init}}$ est déterminé par le modèle directe à partir de $\mathbf{q}_{\mathbf{init}}$).

Dans l'hypothèse de petits déplacements on peut appliquer la méthode de Jacobi :

$$\boldsymbol{x}_{final} = \boldsymbol{f}(\boldsymbol{q}_{final}) = \boldsymbol{f}(\boldsymbol{q}_{init} + \Delta\boldsymbol{q}) \approx \boldsymbol{f}(\boldsymbol{q}_{init}) + \boldsymbol{D}\boldsymbol{f}(\boldsymbol{q}_{init}) * \Delta\boldsymbol{q} = \boldsymbol{x}_{init} + \boldsymbol{D}\boldsymbol{f}(\boldsymbol{q}_{init}) * \Delta\boldsymbol{q}$$

Donc en notant J la jacobienne de f en $\mathbf{q}_{\mathbf{init}}$:

$$\Delta\boldsymbol{x} \approx \mathrm{J} * \Delta\boldsymbol{q}$$

En utilisant les pseudo-inverses de Moore-Penrose, on obtient la solution qui minimise Δq par l'équation :

$$\Delta\boldsymbol{q} \approx \mathrm{J}^{+} * \Delta\boldsymbol{x}$$

Où J^{+} est la matrice pseudo-inverse de J.

On obtient ainsi $\mathbf{q}_{\mathbf{final}}$:

$$q_{final} \approx q_{init} + J^{+} * \Delta x$$

Le calcul pratique de J+ se fait assez simplement dans notre cas. J est représentative d'un système de n équations à m inconnues avec n ≤ m où, sauf cas particulier, les équations sont deux à deux libres, donc le rang de J est égal au nombre de lignes de J. Alors J+ se calcul par la formule :

$$J^{+} = J^{t} * (J * J^{t})^{-1}$$

Où J^t est la transposée de J.

Dans le cas où $(J * J^t)$ n'est pas inversible, cela signifie que deux des lignes de J sont localement liées, il suffit alors de déplacer légèrement la position initiale d'une valeur aléatoire δq telle que $\mathbf{q_{init\ ,\ bis}} = \mathbf{q_{init}} + \boldsymbol{\delta q}$ et de recalculer la jacobienne de f en $\mathbf{q_{init\ ,\ bis}}$.

Il peut arriver qu'une seule itération du calcul donne un résultat numérique $\mathbf{q_{final}}$ qui pour une tolérance **IT** donnée ne vérifie pas $||\mathbf{x_{final}} - \mathbf{f}\ (\mathbf{q_{final}})|| \leq \mathbf{IT}$. Cela est dû à une mauvaise vérification de l'hypothèse des « petits déplacements ». Quelques itérations de l'algorithme (moins de dix) en prenant comme nouvelle position initiale $\mathbf{q_{init\ ,\ k}} = \mathbf{q_{final\ ,\ k\text{-}1}}$ permet d'atteindre la tolérance voulue pour de faibles déplacements.

Dans le **cas d'un déplacement quelconque** on met en place deux nouveaux outils.

Premièrement on défini un « Mapping » d'une dizaine de positions initiales théoriques qui associent des positions $\mathbf{x_k}$ à des valeurs connues des paramètres $\mathbf{q_k}$ vérifiant la relation du modèle directe $\mathbf{x_k} = f\ (\mathbf{q_k})$. Ainsi on choisira parmi les positions du Mapping la valeur initiale de départ $\mathbf{q_i}$ telle que la distance $||\mathbf{x_{final}} - \mathbf{x_i}||$ soit minimale.

Par ailleurs si le tableau de Mapping des positions théoriques de départ est correctement choisi par rapport à l'architecture réelle du système, celui-ci permet d'éviter des valeurs $\mathbf{q_{final}}$ interdites pour le système (cf. collisions entre pièces).

Une fois la position initiale de départ déterminée on découpe le trajet $\mathbf{x_{initial}} - \mathbf{x_{final}}$ en petits déplacements intermédiaires $\mathbf{\Delta x_k} = x_{final\ ,\ k} - \mathbf{x_{init\ ,\ k}}$ pour lesquels on applique l'algorithme précédent.

A l'issu de cet algorithme on obtient donc l'ensemble des paramètres $\mathbf{q_{final}}$ tel que :

$$||x_{final} - f\ (q_{final})|| \leq IT$$

Pilotage réel du robot vers la position finale

Cet algorithme définit l'évolution de la position initiale réelle $\mathbf{q_{init}}$ du système vers la position $\mathbf{q_{final}}$ obtenue en sortie de l'algorithme précédent. Il est extrêmement sommaire, et ne règle pas les problèmes de collision au sain du système de façon exhaustive.

On peut découper le mouvement en n étapes et définir un déplacement élémentaire δq tel que :

$$\delta q = \frac{q_{init} - q_{final}}{n} \quad (\#)$$

Ainsi à l'étape k : $\mathbf{q_k} = k * \boldsymbol{\delta q}$. Cependant cette méthode a de fortes chances d'engendrer des positions interdites pour le système lors du déplacement.

L'algorithme de pilotage doit également éviter les collisions majeures des différentes parties du système entre elles. Ce problème n'est pas étudié dans le détail mais une première approche peut consister à utiliser le Mapping défini pour l'algorithme précédent.

Dans un premier temps on amène le système de sa position initiale à la position du Mapping la plus proche selon le principe de l'équation **(#)**. Avec un mapping bien défini (propre à chaque système), les chances de collisions sont faibles.

Ensuite on définit des trajectoires entre les différentes positions du Mapping qui n'induisent pas de collision. On amène le système grâce à ces trajectoires jusqu'à la position initiale qui a été utilisée pour obtenir $\mathbf{q_{final}}$ lors de l'algorithme de cinématique inverse. Puis on applique à nouveau un découpage homogène du mouvement en utilisant **(#)** entre la position où se trouve maintenant le bras et la position $\mathbf{q_{final}}$.

Chapitre 2 (En cours de rédaction)

APPLICATION AU BRAS DE ROBOT POPPY-HUMANOÏDE

Modèle cinématique directe

Mapping des positions théoriques initiales

Détermination d'une position initiale optimale pour démarrer l'algorithme

Caractérisation d'une trajectoire élémentaire de résolution

Résolution par itération de la méthode de Jacobi

Définition des trajectoires entre les positions du Mapping

Pilotage du déplacement du bras

BIBLIOGRAPHIE

- **Wikipédia**. *Inverse Kinematic*. https://en.wikipedia.org/wiki/Inverse_kinematics, 2015-07-28.
- **Wikipédia**. *Moore-Penrose pseudoinverse*. https://en.wikipedia.org/wiki/Moore-Penrose_pseudoinverse, 2015-07-28.
- **Wikipédia**. *Jacobian matrix and determinant*. https://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant, 2015-07-28.
- **Wikipédia**. *Jacobian inverse technic*. https://en.wikipedia.org/wiki/Inverse_kinematics#The_Jacobian_inverse_technique, 2015-07-28.