TP 3 Robot Dobot: modèle géométrique inverse, traçage d'une droite et dessin d'une figure dans l'espace articulaire

Jean-Louis Boimond
Université Angers

Objectif : Le robot étant muni de l'outil 'Feutre', il s'agit :

- de développer un script à même de tracer un segment de droite sur une feuille, sans utiliser l'application DobotStudio, c'est-à-dire, en utilisant votre propre Modèle Géométrique Inverse (MGI). Pour cela, vous allez :
  - dans la partie A du TP, calculer le MGI du robot en appliquant la méthode de Paul sur le modèle géométrique direct (MGD) calculé lors du TP 2. En déduire un script MatLab qui permette, en fonction des coordonnées (O_{5x}, O_{5y}, O_{5z}) du point $O_5(= PF)$ exprimées dans le repère de base (R_0) du robot, de calculer les coordonnées angulaires correspondantes appliquées au niveau des articulations q_1, q_2, q_3 (angles correspondant à J_1, J_2, J_3 dans l'application DobotStudio) ;
  - dans la partie B du TP, réaliser un script à même de tracer un segment de droite, défini par ses 2 points extrêmes, sur une feuille A4 (posée sur la table à portée du robot), ceci en utilisant votre propre MGI ;

- dans la partie C du TP, d'en déduire un script à même de représenter une figure composée d'un segment de droite et d'un demi-cercle.

A) Calcul du Modèle Géométrique Inverse

1) Calculer le MGI à l'aide de la méthode de Paul pour permettre, en fonction des coordonnées O_{5x}, O_{5y}, O_{5z} du point $O_5(= PF)$ exprimées dans le repère de base (R_0), de calculer les coordonnées articulaires q_1, q_2, q_3 correspondantes (rappel : l'articulation 4 n'est pas contrôlable, l'articulation 5 est inactive).

   Pour cela, vous pourrez vous inspirer du calcul du MGI du robot Stäubli RX 90 fait en Cours (voir §4.6).

   Afin de vous faciliter le calcul à réaliser à partir de « l'Equation de position », à savoir : $\begin{pmatrix} O_{5x} \\ O_{5y} \\ O_{5z} \\ 1 \end{pmatrix} = T_{0,5}(\theta_1, \cdots, \theta_5) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$, les expressions de $T_{1,5} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ et de $T_{2,5} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ seront données.

   Aussi, il va s'agir pour vous :
   - de calculer θ_1, pour en déduire q_1, à partir de la pré-multiplication par $T_{1,0}$ de « l'Equation de position », soit :

$$T_{1,0} \begin{pmatrix} O_{5x} \\ O_{5y} \\ O_{5z} \\ 1 \end{pmatrix} = T_{1,5} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} C_2\,[\mathrm{c}\,C_{3+4} - \mathrm{h}\,S_{3+4} + \mathrm{d}\,C_3 + \mathrm{a}] - S_2\,[c\,S_{3+4} + h\,C_{3+4} + d\,S_3] \\ 0 \\ -S_2\,[\mathrm{c}\,C_{3+4} - \mathrm{h}\,S_{3+4} + \mathrm{d}\,C_3 + \mathrm{a}] - C_2\,[c\,S_{3+4} + h\,C_{3+4} + d\,S_3] \\ 1 \end{pmatrix},$$

   avec : $\mathrm{a} = 135, \mathrm{c} = 41 + 20(= 61), \mathrm{d} = 147, \mathrm{h} = 82.7$ (valeurs en *mm*) et sachant que C_i, resp. S_i, est l'abréviation de $\mathrm{Cos}(\theta_i)$, resp. $\mathrm{Sin}(\theta_i)$.

Comme pour la méthode utilisée en Cours (§4.6), vous pourrez calculer θ_1 à partir de la 2ème équation du système précédent (constitué de 4 équations).
Vous noterez que la butée mécanique de l'angle q_1 (décrite à la page 2 du TP 1) est telle que seule une solution est possible au sens où le robot ne peut pas « se retourner » de 180° sur son axe $\overrightarrow{O_0 z_0}$ (il n'est pas possible d'atteindre des valeurs négatives selon l'axe $\overrightarrow{O_0 x_0}$).

- de calculer θ_2, puis θ_3, pour en déduire q_2 et q_3, à partir de la pré-multiplication par $T_{2,0}$ de « l'Equation de position », soit :

$$T_{2,0}\begin{pmatrix} O_{5x} \\ O_{5y} \\ O_{5z} \\ 1 \end{pmatrix} = T_{2,5}\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \mathrm{c}\, C_{3+4} - \mathrm{h}\, S_{3+4} + \mathrm{d}\, C_3 + \mathrm{a} \\ \mathrm{c}\, S_{3+4} + \mathrm{h}\, C_{3+4} + \mathrm{d}\, S_3 \\ 0 \\ 1 \end{pmatrix}.$$

Comme pour la méthode utilisée en Cours (voir §4.6), simplifier, dans un premier temps :
- l'expression de $T_{2,0}\begin{pmatrix} O_{5x} \\ O_{5y} \\ O_{5z} \\ 1 \end{pmatrix}$ en posant $b_1 = C_1 O_{5x} + S_1 O_{5y}$,
- l'expression de $T_{2,5}\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ sachant que $\theta_3 + \theta_4 = -\theta_2$.

A partir du système d'équations qui en résulte (constitué de 3 égalités), extraire 2 équations, l'une exprimant C_3, l'autre S_3, en fonction de C_2, S_2.
Le fait d'ajouter ces 2 équations, préalablement mises au carré, permet de disposer d'une équation de la forme :

$$X\, Sin(\theta_2) + Y\, Cos(\theta_2) = Z$$

avec : $X = 2a(O_{5z} + h)$, $Y = -2a(b_1 - c)$ et $Z = d^2 - (b_1 - c)^2 - (O_{5z} + h)^2 - a^2$,

et où les 2 solutions sont décrites dans le Cours (voir §4.6).

En déduire les valeurs de θ_2, puis θ_3 (sachant que vous disposez de l'expression de C_3 et S_3 en fonction de C_2, S_2). Vous noterez que les butées mécaniques des angles q_2, q_3 sont telles que seule la posture « coude haut » est possible (la posture « coude bas » ne l'est pas).

2) En déduire un script, sous la forme d'une fonction MatLab (voir instruction function), qui permette le calcul du MGI. **Veiller** à ce que la fonction restitue les angles q_1, q_2, q_3 en radian (et non en degré) afin de pouvoir être utilisée par la suite (dans les parties B.3.b, B.3.c, C). Vérifier l'exactitude de votre script en appliquant, sur quelques exemples, les valeurs articulaires issues du MGI à l'entrée du MGD et, ainsi, vérifier que vous retrouvez bien les coordonnées appliquées à l'entrée du MGI. Utiliser DobotStudio (pour cela, installer la **version 1.9.4** de DobotStudio en vous aidant des indications données dans la partie III (p.5) du TP1 présentant le robot) pour relever ces coordonnées en pensant à sélectionner au préalable l'outil Advanced avec les valeurs configurées comme suit :

EndType : PF
$xBias = 61; yBias = 0; zBias = 82{,}7$

B) Traçage d'une droite

Etude du mouvement en mode Linéaire (dans l'espace opérationnel) et Joint (dans l'espace articulaire) :

Soient les points $P_1 = (214; -117; -138)$ et $P_2 = (227; 91; -138)$ (valeurs en *mm*) définis dans le repère de base (R_0) du robot.

1) Montrer que ces 2 points appartiennent à l'espace d'atteignabilité du bras du robot et vont bien pouvoir être représentés sur une feuille A4 disposée sur la table. Pour rappel, la figure suivante indique les dimensions du bras du robot et du dispositif de fixation des outils.

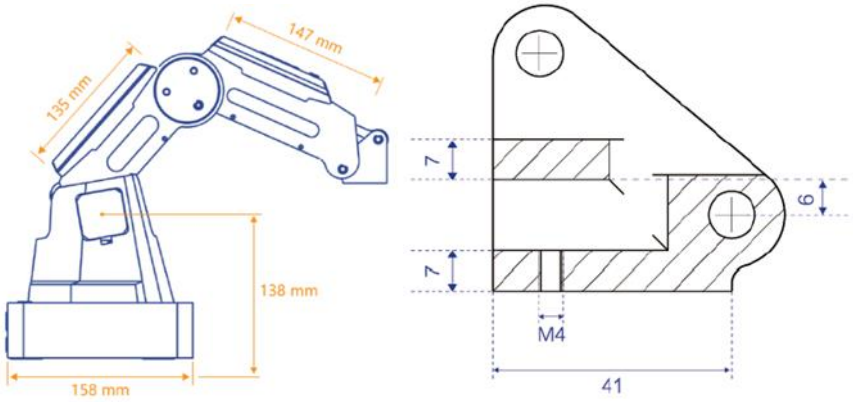

Figure 1 : dimensions du bras du robot et du dispositif de fixation des outils.

N.B. : Dans la suite du TP, **sélectionner l'outil « Advanced »** avec les valeurs préalablement configurées comme suit :

EndType : PF
$xBias = 61; yBias = 0; zBias = 82{,}7$

ceci pour que les valeurs *X*, *Y*, *Z* affichées dans l'encadré « Operation Panel » de DobotStudio désignent les coordonnées de la pointe du Feutre définies dans le repère de base (R_0) du robot.

2) *Via* la fonctionnalité 'Bockly' de DobotStudio (voir documentation du TP 1), il s'agit de tracer un segment de droite entre les points P_1, P_2 en utilisant le mode Linéaire dans l'espace opérationnel (voir instruction MoveTo X Y Z décrite dans la documentation du TP 1). Pour cela, après avoir calibré le robot (instruction Home), aller en utilisant le mode Linéaire (dans l'espace opérationnel) : au point situé à 8 *mm* à la verticale du point P_1 (à savoir les coordonnées $(214; -117; -130)$), puis au point P_1, puis au point P_2, puis au point situé à 8 *mm* du point P_2 (à savoir les coordonnées $(227; 91; -130)$), puis au point correspondant à la configuration initiale du robot (*i.e.*, lorsque tous les angles sont nuls au niveau des articulations).

3) Le fait d'utiliser le mode Linéaire dans l'espace opérationnel nécessite l'emploi du MGI codé dans DobotStudio. On veut à présent s'affranchir de ce modèle inverse en traçant cette droite en mode Joint, c'est-à-dire, dans l'espace articulaire (où la commande à appliquer est directement calculée au niveau des articulations).

   a) Au préalable, observons ce que permet le mode Joint dans l'espace articulaire *via* la fonctionnalité 'Teaching & Playback' de DobotStudio (voir documentation du TP 1). Pour cela, après avoir calibré le robot, aller en utilisant le mode Joint (voir instruction MOVJ décrite dans la documentation du TP 1) : au point situé à 8 *mm* à la verticale du point P_1, puis au point P_1, puis au point P_2, puis au point situé à 8 *mm* du point P_2, puis au point correspondant à la configuration initiale du robot. Qu'observez-vous sur le comportement de la trajectoire comparativement à celui généré dans la question B.2 ?

b) Vérifier si la trajectoire précédente correspond à/(est proche de) une trajectoire issue d'une commande basée sur une interpolation faite au niveau des articulations (en mode Joint dans l'espace articulaire).
Pour cela, il y a 3 parties à compléter (situées aux lignes 27, 34 et 42) dans le script Matlab « Q_B_3_b_A_completer.m » (ce script, ainsi que ceux nécessaires à son exécution, sont contenus dans le fichier zip accessible *via* le lien Tools_TP3.zip) afin de réaliser une interpolation au niveau des articulations en calculant :

- *via* le MGI les vecteurs articulaires (de dimension 3×1) correspondant aux points P_1, P_2 (points extrêmes de la trajectoire à tracer),
- les 9 points (vecteurs de dimension 3×1) intermédiaires répartis de manière régulière sur chacune des 3 droites délimitées par les vecteurs articulaires correspondants aux points P_1, P_2.

Ces 11 coordonnées articulaires (à savoir, celle correspondant au point P_1, les 9 points intermédiaires, celle correspondant au point P_2) vont être appliquées à la commande du robot à l'issue des instructions suivantes :

- une connexion au robot *via* la fonction MatLab : FCT_connecter,
- l'activation du Buffer (géré en FIFO) interne au contrôleur du robot afin d'exécuter les commandes que l'on souhaite appliquer. Pour cela, le Buffer est d'abord vidé *via* l'API : SetQueuedCmdClear ; puis l'ordre d'exécuter (dans l'ordre de leurs arrivées) les commandes placées dans le Buffer est activé *via* l'API : SetQueuedCmdStartExec,
- l'envoi dans le Buffer d'une commande HOME afin de calibrer le bras du robot *via* la fonction MatLab : FCT_cmd_HOME,
- l'envoi dans le Buffer des commandes de mouvement en mode Joint des 11 coordonnées (dans l'ordre, celle correspondant au point P_1, les 9 coordonnées calculées par interpolation, celle correspondant au point P_2) *via* la fonction MatLab : FCT_cmd_PTP.

Une dernière commande de mouvement est envoyée dans le Buffer pour permettre au robot de rejoindre la configuration initiale. Une fois effectuée l'exécution de la dernière commande (attente réalisée *via* la fonction MatLab : FCT_indice_courant_queue), l'ordre d'exécution des commandes placées dans le Buffer est désactivé *via* l'API : SetQueuedCmdStopExec ; puis le robot est déconnecté *via* l'API : DisconnectDobot.

c) En déduire une solution à même de tracer une ligne quasiment droite entre les points P_1 et P_2 en utilisant une méthode d'interpolation pour calculer 18 points intermédiaires. Que pensez-vous de la précision de la ligne tracée ? Comment l'améliorer ?

Proposer un script MatLab à même de tracer la ligne droite en reprenant une démarche similaire à celle utilisée dans B.3.b.

C) Dessin d'une figure dans l'espace articulaire

Proposer un script MatLab inspiré de celui réalisé dans B.3.c à même de représenter la figure suivante composée d'un segment de droite parallèle à l'axe $\overrightarrow{y_0}$ du repère de base (R_0) du robot et d'un demi-cercle de rayon 4 *cm* dont le centre 'C' a pour coordonnées $(220;\ 0; -138)$ dans le repère R_0. Vous pourrez, par exemple, utiliser 20 points pour tracer le segment de droite et 30 points (calculés en utilisant la représentation polaire du cercle) pour tracer le demi-cercle.

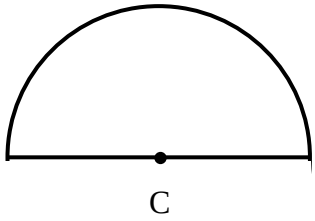

Figure 2 : demi-cercle ‘supérieur’ à reproduire.