Génération de trajectoires dans l'espace de la tâche avec un robot manipulateur

I Objectif

Le but de ce projet est de développer une primitive de mouvement afin de faire réaliser à l'organe terminal d'un robot manipulateur un mouvement imposé (géométrie et vitesse).
La trajectoire dans l'espace opérationnel se fait à vitesse imposée et les modèles d'un robot manipulateur permettent de calculer les commandes en position, vitesse et accélération à envoyer au robot pour générer le mouvement désiré.
La tâche est donc définie dans l'espace opérationnel et il est demandé de mettre en évidence le lien entre l'espace opérationnel et l'espace généralisé. Travail à réaliser :

- calculer les différents modèles,
- calculer les trajectoires $\boldsymbol{X}(\mathrm{t})$, $\dot{\boldsymbol{X}}(\mathrm{t})$et $\ddot{\boldsymbol{X}}(\mathrm{t})$ à chaque instants d'échantillonnage,
- calculer les trajectoires $\boldsymbol{q}(\mathrm{t})$, $\dot{\boldsymbol{q}}(\mathrm{t})$, $\ddot{\boldsymbol{q}}(\mathrm{t})$ à chaque instants d'échantillonnage,
- simuler vos trajectoires.

Ce BE est découpé en 4 séances de Tps encadrées (3h, 3h, 4h, 2h).

Evaluation : chaque groupe doit rendre un rapport ainsi que le code à la fin du projet. Une évaluation partielle est faite à chaque séance et complétée ensuite par des tests après la livraison finale.

Temps estimé : 12 h en présence des encadrants (4 séances) et 8h-12h hors encadrement.
Il est fortement conseillé de se répartir le travail au sein du groupe afin d'optimiser le temps de travail et de ne pas déborder du volume horaire indiqué.
La première partie du travail concerne le calcul du MGD du robot $UR3$.

Pour la suite du BE, afin de simplifier les calculs, on supposera que les 3 dernières liaisons sont fixes afin de se ramener à un robot 3R, ce UR3 réduit noté $UR3R$ sera notre robot de travail.

II Modélisation géométrique directe du robot UR3

L'UR3 d'Universal Robot est un robot 6R. La figure 1 vous indique ses axes de rotations ainsi que ses dimensions nécessaires pour établir les modèles.
La configuration du robot est définie dans l'espace généralisé par $\boldsymbol{q} = (q_1, q_2, q_3, q_4, , q_5, q_6)^T$.
On ne considère pas de butée mécanique sur ce robot dans un premier temps, i.e., $\boldsymbol{q} \in \mathbb{R}^6$.
La situation de l'O.T. est définie par la position du point O_e et par l'orientation du repère de l'effecteur R_e par rapport au repère de base R_0, (O_0, x_0, y_0, z_0).

Travail

Sachant que l'origine du repère 3 (O_3) et que le repère outil (noté $_e$) sont imposés (voir figure 2), calculer le MGD de ce robot.

- En vous servant des données techniques indiquer les dimensions manquantes sur la figure 2.
- Placer les repères et déterminer les paramètres de DHM associés (figure 2).
- Programmer une fonction Python qui calcule votre MGD de manière numérique (produit des matrices élémentaires). Tester votre MGD.

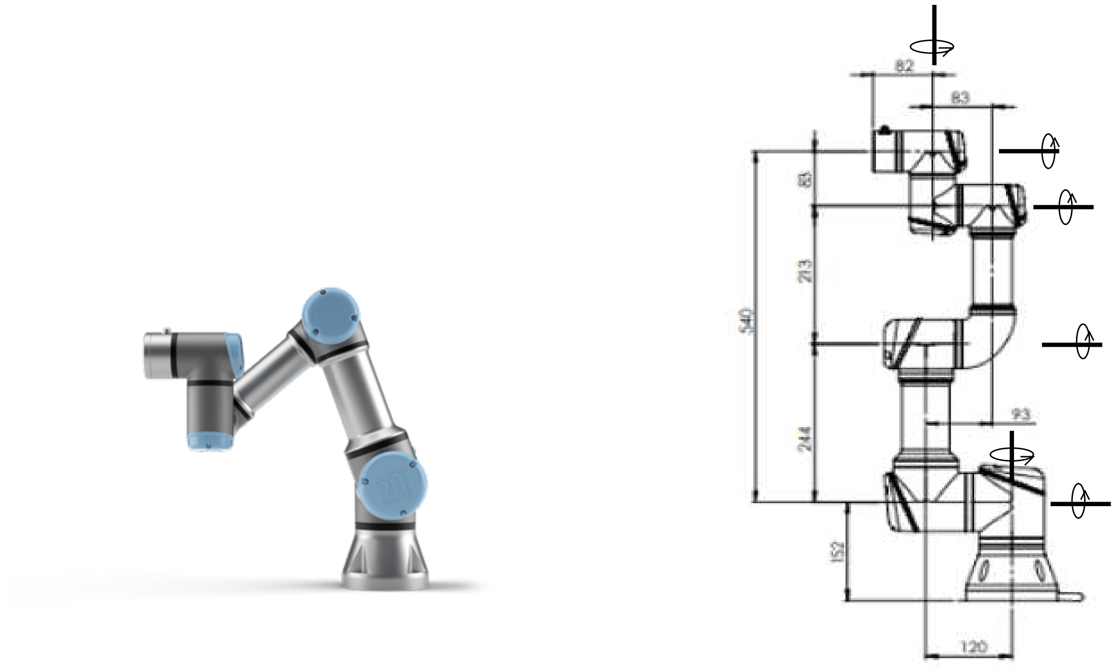

FIGURE 1 – Dimensions géométriques de l'UR3

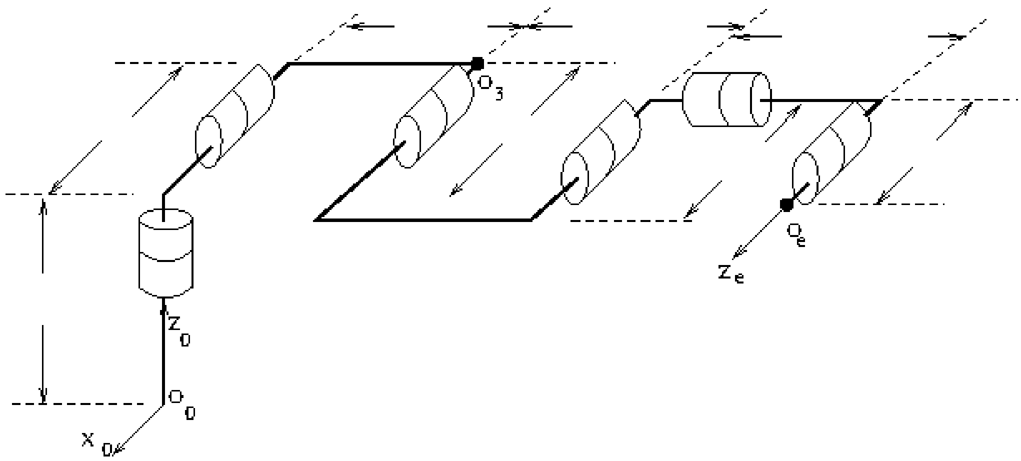

FIGURE 2 – Modélisation des liaisons de l'UR3

III Modélisation géométrique du robot simplifié

Afin de simplifier le calcul des modèles, on considère que les 3 dernières liaisons sont fixes dans la configuration de la figure 2.
On a donc la configuration du robot qui est définie par $\boldsymbol{q} = (q_1, q_2, q_3)^T$ et le vecteur de situation est défini par $\boldsymbol{X} = (x_e, y_e, z_e)^T$.
Le robot est schématisé sur la figure 3. Ce robot **UR3R** est le robot avec lequel on va faire la génération de mouvement.

Travail

- Calculer le MGD analytique de l'UR3R.
- Implémenter en Python votre fonction MGD analytique.
- Calculer le MGI connaissant la position du point $X_e = (x_e, y_e, z_e)$. Ici on utilise seulement les équations fonction de la position du point O_e

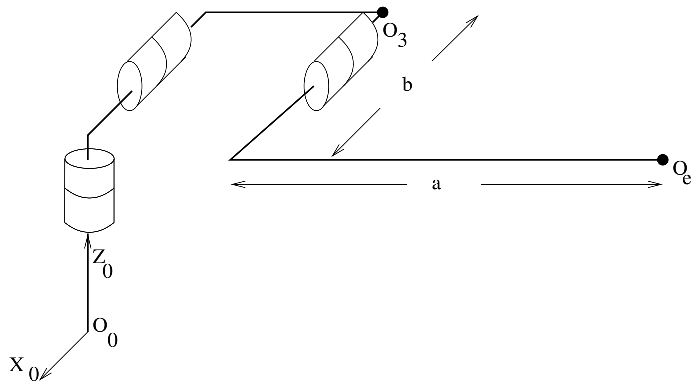

FIGURE 3 – Modélisation des liaisons du UR3R

- Implémenter votre calcul du MGI.
- Tester vos fonctions.

IV Modélisation différentielle

- Calculer la jacobienne (géométrique) de l'UR3R.
- Calculer la jacobienne analytique permettant de connaître $\dot{\boldsymbol{X}}_{\boldsymbol{e}} = (\dot{\boldsymbol{x}}_e, \dot{\boldsymbol{y}}_e, \dot{\boldsymbol{z}}_e)^T$.
- Implémenter en Python votre fonction MDD.
- Tester vos fonctions jacobiennes.

V Génération de mouvement

V.1 Cahier des charges

Développer une primitive de mouvement qui fait effectuer à l'OT une trajectoire rectiligne entre deux points 3D, A et B, avec des vitesses imposées et connues à chaque $t = k \cdot T, k = 0, 1, 2, 3, 4$.
La figure 4 montre les profils de vitesse et d'accélération de l'OT entre A et B.
Toutes les accélération/décélération se font à valeur constante K_1 et K_2.
On note $traj(A, B, V_1, V_2)$ cette primitive qui doit retourner les vecteurs $\boldsymbol{q}$(t) , $\dot{\boldsymbol{q}}$(t) et $\ddot{\boldsymbol{q}}$(t) .

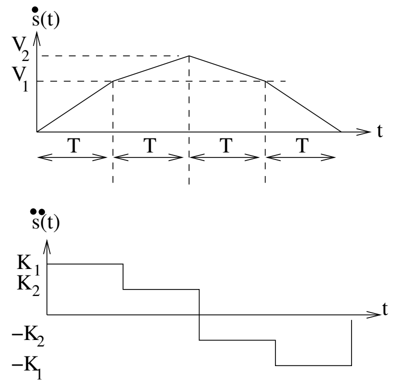

FIGURE 4 – Profils de vitesse et d'accélération de l'abscisse curviligne

Sachant que vous devez envoyer les consignes au robot avec une période Te (de l'ordre de 1 à 10 ms) vous devez calculer vos lois à chaque instants d'échantillonnage.

V.2 Trajectoire opérationnelle $\boldsymbol{X}$(s) de type segments de droite

La trajectoires est rectiligne entre les points $(A,\ B)$.
$(x_i,\ y_i, z_i)$ représentent respectivement les coordonnées du points i dans le repère $(O_0,\ x_0,\ y_0, z_0)$.

- Calculer les coordonnées $(x(s)\ ,\ y(s)\ ,\ z(s)\ )$ des points du segment de droite en fonction de s.
- Calculer les vitesses et les accélérations opérationnelles $(\dot{x}(s),\ \dot{y}(s),\ \dot{z}(s)),\ (\ddot{x}(s),\ \ddot{y}(s),\ \ddot{z}(s))$.
- Programmer le calcul de la trajectoire opérationnelle.
- Afficher la trajectoire opérationnelle.

V.3 Loi de mouvement temporelle

Déterminer les lois d'évolution $s(t)$, $\dot{s}(t)$, $\ddot{s}(t)$ de telle sorte que la trajectoire respecte le profil de vitesse imposé (figure 4) La vitesse de l'organe terminal, O_E, initialement nulle en A, atteint le point B avec une vitesse nulle.

- Programmer le calcul de la loi de mouvement $s(t)\ ,\ \dot{s}(t)\ ,\ \ddot{s}(t)$.
- Afficher les courbes correspondantes.

V.4 Génération de mouvement dans l'espace de la tâche $\boldsymbol{X}$(t)

- Connaissant $s(t)$, $\dot{s}(t)$, $\ddot{s}(t)$ et $x(s)\ ,\ y(s)\ ,\ z(s)\ ,\ \dot{x}(s)\ ,\ \dot{y}(s)\ ,\ \dot{z}(s)\ ,\ \ddot{x}(s)\ ,\ \ddot{y}(s)\ ,\ \ddot{z}(s)$ afficher $\boldsymbol{X}$(t), $\dot{\boldsymbol{X}}$(t), $\ddot{\boldsymbol{X}}$(t).
- Calculer et afficher la vitesse du point O_E.

V.5 Génération de mouvement dans l'espace articulaire (généralisé)

- A l'aide des modèles inverses, calculer $\boldsymbol{q}$(t) , $\dot{\boldsymbol{q}}$(t) , $\ddot{\boldsymbol{q}}$(t) (entrées de la commande)
- Programmer votre fonction $traj(A, B, V_1, V_2)$ qui prend en entrée les coordonnées des points, les vitesses désirées de l'outil et qui retourne $\boldsymbol{q}$(t) , $\dot{\boldsymbol{q}}$(t) , $\ddot{\boldsymbol{q}}$(t) .
- Tester votre programme dans différents cas pour s'assurer de la robustesse de votre code.

Rappel : Modélisation
Dans le but de simuler une trajectoire dans l'espace généralisé à partir d'une trajectoire dans l'espace opérationnel, il est nécessaire de calculer les différents modèles inverses.

Modèles géométriques	Modèles cinématiques	Modèles d'accélération
$\boldsymbol{X}$(t)= F ($\boldsymbol{q}$ (t))	$\dot{\boldsymbol{X}}= J\ \dot{\boldsymbol{q}}$	$\ddot{\boldsymbol{X}}= J\ \ddot{\boldsymbol{q}}+ \dot{J}\ \dot{\boldsymbol{q}}$
$\boldsymbol{q}$ (t) = F^{-1} ($\boldsymbol{X}$(t))	$\dot{\boldsymbol{q}}= J^{-1}\ \dot{\boldsymbol{X}}$	$\ddot{\boldsymbol{q}}= J^{-1}\ (\ \ddot{\boldsymbol{X}}\text{-}\ \dot{J}\ \dot{\boldsymbol{q}})$

Rappel : Génération de mouvement
De manière générale, la génération de mouvement dans l'espace opérationnel s'effectue en deux étapes : tout d'abord, on définit la trajectoire géométrique désirée pour l'organe terminal du robot, $\boldsymbol{X}$(s) , avec s l'abscisse curviligne le long de la trajectoire. Puis on établit une loi d'évolution temporelle, $s(t)$, sur cette trajectoire, cette dernière permettant de caractériser le mouvement. On obtient $\boldsymbol{X}$(s(t))= $\boldsymbol{X}$(t).