Commande cinématique d'un robot UR3

I. Objectifs de la séance

Lors de cette séance, vous implémenterez une loi de commande cinématique qui permet à l'organe terminal d'un robot d'atteindre une attitude désirée. Vous considérerez que vous disposez des modèles géométrique et cinématique du robot. Vous ferez également l'hypothèse que les boucles de contrôle locales en vitesse possèdent une bande passante très supérieure à celle de la tâche opérationnelle à accomplir.

II. Environnement de travail

Le travail se fera en simulation au sein d'un environnement de développement Python et avec l'aide de la libraire Robotics Toolbox développée par Peter Corke [1].

Vous vous intéresserez plus particulièrement, et sans perte de généralité, au robot UR3 à 6 degrés de liberté, et nous mettrons en œuvre une loi de commande basée sur la représentation axe angle. Les axes de l'UR3 sont asservis en vitesse. Au vu de l'hypothèse énoncée plus haut, vous considérerez que les entrées de commande articulaire seront directement égales aux vitesses articulaires.

III. Travail à effectuer

A. Travail préliminaire de modélisation

De façon générale, toute rotation d'un angle θ autour d'un vecteur unitaire $u = [u_x \ u_y \ u_z]'$ admet une matrice de rotation de la forme suivante :

$$\begin{aligned}\mathrm{R}(\theta, \mathrm{u}) &= \begin{pmatrix} u_x^2(1-\cos(\theta)) + \cos(\theta) & u_x u_y(1-\cos(\theta)) - u_z \sin(\theta) & u_x u_z(1-\cos(\theta)) + u_y \sin(\theta) \\ u_x u_y(1-\cos(\theta)) + u_z \sin(\theta) & u_y^2(1-\cos(\theta)) + \cos(\theta) & u_y u_z(1-\cos(\theta)) - u_x \sin(\theta) \\ u_x u_z(1-\cos(\theta)) - u_y \sin(\theta) & u_y u_z(1-\cos(\theta)) + u_x \sin(\theta) & u_z^2(1-\cos(\theta)) + \cos(\theta) \end{pmatrix} \\ &= \cos(\theta)\mathbb{I}_3 + \sin(\theta)\mathrm{AS}(\mathrm{u}) + (1-\cos(\theta))\mathrm{uu}'\end{aligned}$$

Dans le cours nous avons vu que (θ, u) peuvent être obtenus à partir de la matrice de rotation de la sorte :

$$(\theta, u) = \begin{cases} \theta = \arccos\left(\dfrac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \\ \mathbf{u} = \frac{1}{2\sin(\theta)} \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix} \end{cases} \qquad \text{ou encore } \theta\mathbf{u} = \frac{\theta}{2\sin(\theta)} \underbrace{\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}}_{l} \tag{1}$$

Comme vous pouvez le voir, la représentation axe/angle admet des singularités pour $\theta = k\pi$

1) Etablissez la relation entre la trace de la matrice de rotation $\mathrm{R}(\theta, \mathrm{u})$ et $\cos(\theta)$
2) Calculez successivement le vecteur l pour $\theta = 0$ et $\theta = \pi$ qui correspondent à un facteur près aux angles de singularité de la représentation axe/angle
3) Calculez les traces de la matrice $\mathrm{R}(\theta, \mathrm{u})$ et $\cos(\theta)$ pour $\theta = 0$ et $\theta = \pi$. Donnez à chaque fois le signe de la trace

1. P.I. Corke : https ://github.com/petercorke/robotics-toolbox-python/wiki

4) Ecrivez maintenant les matrices de rotation $\mathrm{R}(\pi, \mathrm{u_x})$, $\mathrm{R}(\pi, \mathrm{u_y})$ et $\mathrm{R}(\pi, \mathrm{u_z})$

5) Montrer que la norme de l est égale à $2\sin(\theta)$

6) Comme $\theta = 0$ et $\theta = \pi$ sont des singularités pour la représentation axe/angle. Proposez des solutions qui permettent de les contourner.

B. Etude de la stabilité numérique

Lors de l'implémentation d'une loi de commande basée sur le paramétrage axe/angle, vous commencerez par le calcul de la matrice de rotation qui correspond à l'erreur avant d'en extraire l'axe et l'angle.

Dans cette partie, vous allez procéder à l'envers : le script « part_1.py » vous donne la possibilité de définir un vecteur de rotation, qui sera normalisé ainsi qu'un angle de rotation.

1) Commencez par lire le code et complétez-le.
2) Modifier la valeur de l'angle en gardant une rotation autour d'un angle différent de $k\pi$ et vérifier la cohérence des résultats avec la préparation
3) Saisir à tour de rôle $\theta = 0$ et ensuite $\theta = \pi$. Commentez.

Ouvrez maintenant le script "part_2.py". Comme dans le script précédent, vous allez pouvoir définir un axe de rotation unitaire. Le code conduit ensuite une comparaison d'un angle θ qui varie dans $]-\pi, \pi[$ avec son estimation en utilisant l'arc-cosinus telle qu'exprimée dans l'équation (1) et l'arc-tangente 2.

1) Commencez par lire le code et complétez-le en fonction des résultats obtenus plus haut
2) Ajouter 10^{-4} à la variable arg et observez les commentaires générés lors de l'interprétation du script.

C. Commande opérationnelle

Vous allez maintenant vous attaquer à la commande opérationnelle à proprement parler. Pour cela ouvrez les scripts "tp_main.py" et "commande.py".

1) Commencez par lire le code de "tp_main.py" afin d'en saisir la structure.
2) Lisez le code contenu dans "commande.py" pour le comprendre et le compléter
3) Dès que la loi de commande donne des résultats satisfaisants, testez les cas de singularité
4) Modifier le gain proportionnel et commentez.