ROB315 - MODÉLISATION ET COMMANDE DES ROBOTS MANIPULATEURS

TP n°1 Cinématiques Directe et Inverse et TP n°2 Dynamique et Commande

1er février 2020

Gabriel Henrique Riqueti

Victor Kenichi Nascimento Kobayashi

ENSTA IP Paris

Gabriel Henrique Riqueti
Victor Kenichi Nascimento Kobayashi ENSTA IP Paris

TP n°1 - Cinématiques Directe et Inverse

Modèle géométrique direct

Question 1

À partir de la Figure 1 nous pouvons voire les repères associés aux corps du robot selon la convention DHM.

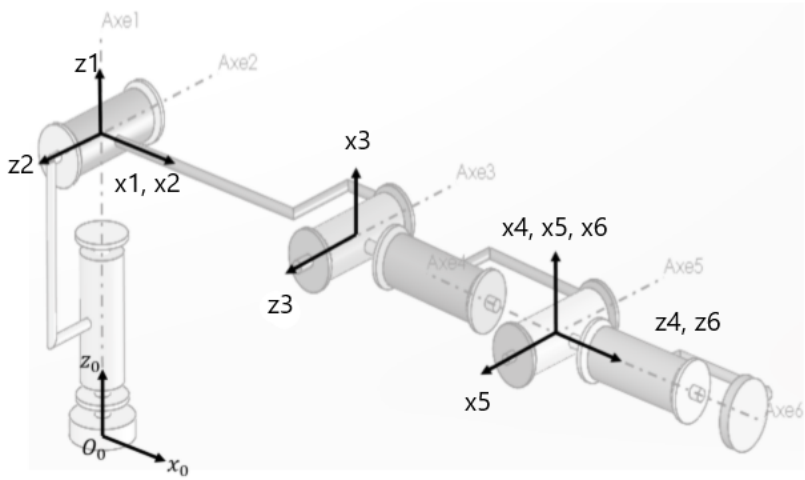

FIGURE 1: Repères attachés aux corps du robot 6R.

Selon la convention de Denavit-Hartenberg Modifiée, dite de Khalil-Kleinfinger, on a écrit la fonction `VisualisationChaine` qui affiche la structure du robot manipulateur 6R. On peut la tester en lançant le fichier `Q1`. On note clairement qui on a réussi a bien implémenté la représentation des repères du robot.

Question 2

En analysant la Figure 1 et en s'appuyant sur la convention de *Khalil-Kleinfinger* nous avons rempli le Tableau 1. Afin d'utiliser toujours une orientation en conformité avec cette convention, on a écrit la fonction `InitValuesTP1` qui remplit cette matrice donnée le vecteur

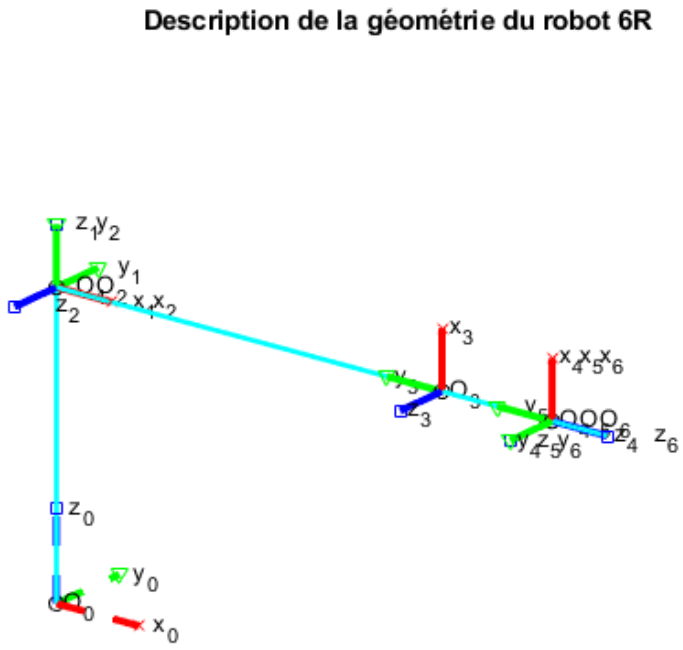

FIGURE 2: Description de la géométrie du robot 6R.

de coordonnées articulaires.

i	α_i	d_i	θ_i	r_i
1	0	0	q_1	r_1
2	$\pi/2$	0	q_2	0
3	0	d_3	$q_3+\pi/2$	0
4	$\pi/2$	0	q_4	r_4
5	$-\pi/2$	0	q_5	0
6	$\pi/2$	0	q_6	0
E	0	0	0	r_E

Tableau 1: Paramètres géométriques du robot

Question 3

Cette question est divisée en 3 parties. D'abord nous avons écrit une fonction pour calculer la matrice de transformation homogène qui est sur le fichier `CalculTransformationElem.m`qui

retourne la matrice homogène $\bar{\mathbf{g}}$ entre deux repères consécutifs.

Ensuite, nous avons fait une fonction pour calculer la matrice de transformation qui est sur le fichier `CalculMGD.m`qui calcule le Modèle Geometrique Direct (MGD) d'un robot quelconque en chaîne ouverte simple prenant comme argument d'entrée les vecteurs de paramètres géométriques du robot (α, d, θ, r).

Finalement, à partir des résultats de la question 2, nous faisons un script définit par `Q03.m` pour calculer la matrice de transformation homogène g_{0E} donnant la position et l'orientation du repère terminal R_E attaché à l'organe terminal du robot dans le repère de base R_0 pour le cas dont les coordonnées articulaires sont nulles.

La matrice de transformation homogène obtenue est montré ci-dessous :

$$g_{0E} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Question 4

Les valeurs des positions $P_x, P_y et P_z$ bien comme le vecteur directeur et l'angle de rotation affichés dessous sont calculés par le fichier `Q04Test.m` qui utilise le script `AxeAngleRot.m` pour calculer le vecteur directeur et l'angle de rotation.

$$P_i = \begin{bmatrix} -0.1 \\ -0.7 \\ 0.3 \end{bmatrix} \qquad P_f = \begin{bmatrix} 0.6364 \\ -0.1 \\ 1.1364 \end{bmatrix}$$

$$n_i = \begin{bmatrix} -0.5774 \\ -0.5774 \\ 0.5774 \end{bmatrix}$$

$$n_f = \begin{bmatrix} 0.2811 \\ 0.6786 \\ -0.6786 \end{bmatrix}$$

$q_i = 2.0994$ $q_f = 2.5936$

Question 5

On a écrit la fonction `VisualisationRepere.m` pour l'affichage des repères de la base (en noir) et de l'organe terminal (en vert). Pour la tester, il faut lancer le fichier `Q5Test.m` qui ouvre la Figure 3.

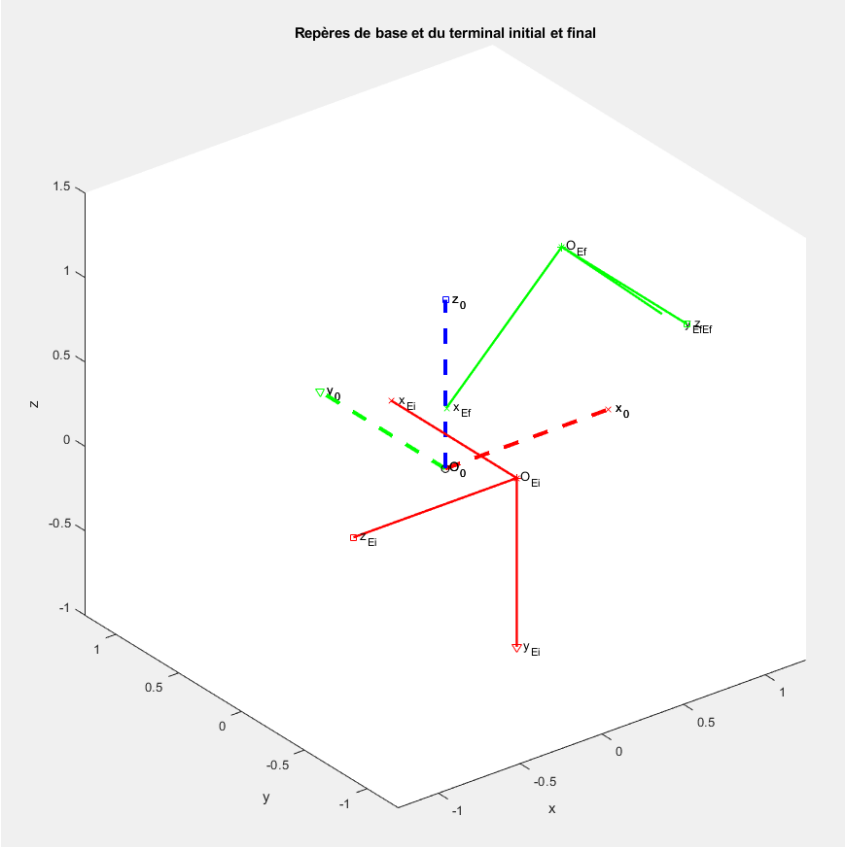

FIGURE 3: Visualisation des repères de base 0 en noir et terminal E en vert.

Modèle cinématique direct

Question 6

Le fichier `CalculJacobienne.m` contient la fonction qui retourne la matrice jacobienne calculée par la méthode des composition de vitesses.

Les valeurs de torseurs cinématiques selon les conditions donnés sont affichés ci-dessous :

$$dXE_i = \begin{bmatrix} 0.3500 \\ -0.1000 \\ 0.6000 \\ -0.0000 \\ -1.0000 \\ 0.0000 \end{bmatrix} \qquad dXE_f = \begin{bmatrix} -0.5510 \\ 0.3182 \\ 0.4596 \\ 1.0607 \\ -0.0000 \\ 0.1464 \end{bmatrix}$$

Question 7

On a écrit la fonction `ChampdeVitesse` pour l'affichage des ellipsoïdes de vitesse du organe terminal pour les configurations q_i et q_f. Pour la tester, il faut lancer le fichier `Q07Test.m` qui ouvre la Figure 4.

En examinant les vecteurs singulières à gauche de la matrice Jacobienne en prenant en compte seulement les lignes de vitesse multiplié par sa transposée, on obtient les trois directions du ellipsoïd de vitesse. La première colonne corresponde à direction de la première valeur singulière qui est la plus importante. Alors, les directions privilégiées sont les suivantes :

$$U_i = \begin{bmatrix} -0.0140 \\ -0.7016 \\ -0.7124 \end{bmatrix} \qquad U_f = \begin{bmatrix} -0.0145 \\ 0.7018 \\ 0.7122 \end{bmatrix}$$

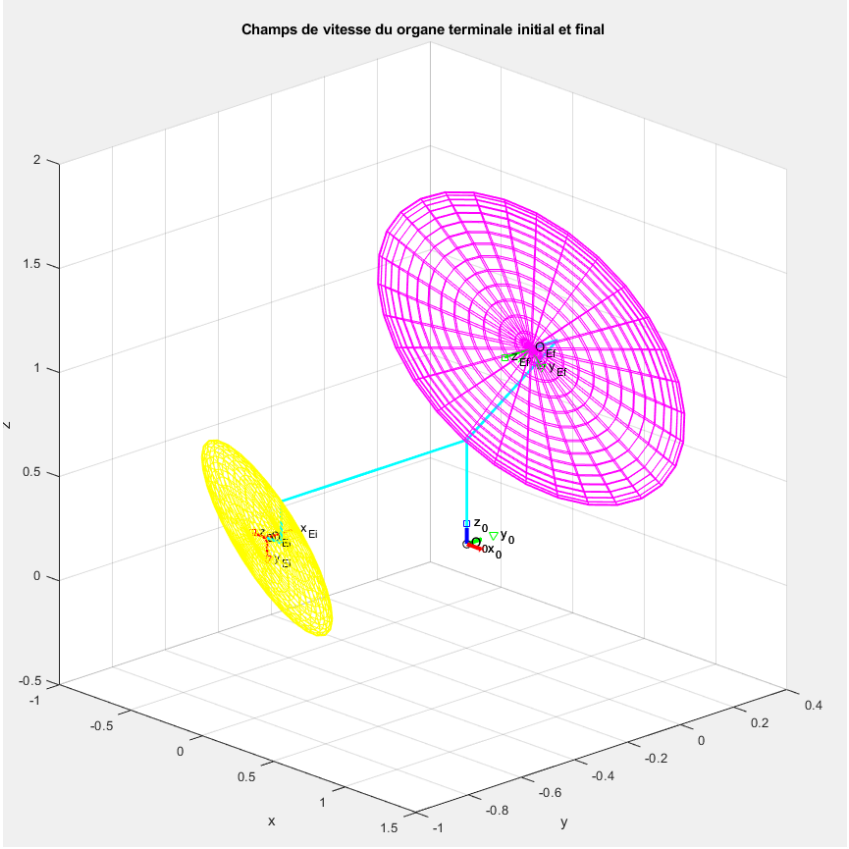

FIGURE 4: Ellipsoïdes de Vitesse correspondant aux configurations q_i et q_f.

Concernant la manipulabilité en vitesse, ils valent $W_i = 0,0124$ $W_f = 0,0035$ dans les configurations q_i et q_f respectivement.

Modèle géométrique inverse

Question 8

On a écrit la fonction `MGI` pour trouver les coordonnées articulaires données la position du organe terminal. Pour la tester pour les configurations q_i et q_f, il faut lancer le fichier `Q08Test.m`.

$$q*_i = \begin{bmatrix} -1.5710 \\ -0.0021 \\ -1.4711 \\ -1.4695 \\ -1.4705 \\ -1.4700 \end{bmatrix}$$

$$Xd*_i = \begin{bmatrix} -0.0991 \\ -0.7113 \\ 0.2915 \end{bmatrix}$$

L'erreur a été calculé selon la norme entre X_d et $X_d i$. Les valeurs données par l'item 1 nous avons obtenu un erreur euclidienne de 0,0142, un peu plus petit que l'erreur initial de 0,0151 m. Alors, l'algorithme n'atteint pas à la tolérance désirée de 0,001 m dans 100 itérations.

$$q*_f = \begin{bmatrix} -0.0037 \\ 0.7770 \\ -0.0052 \\ 1.0025 \\ 1.9981 \\ 0 \end{bmatrix} \qquad Xd*_f = \begin{bmatrix} 0.5783 \\ -0.0788 \\ 1.1365 \end{bmatrix}$$

Avec es valeurs données par l'item 2, nous avons obtenu un erreur de 0.0653 m aussi un peu plus petit que l'erreur initial de 0,0813 m. Alors, l'algorithme n'atteint pas à la tolérance désirée de 0,001 m à nouveau.

Ces erreurs sont de l'ordre de centimètres, ce qui est très élevé. Une solution est augmenter

le nombre d'itérations qui cependant augmente le temps d'exécution de calcul et autrement, on peut augmenter le pas de mise à jour du méthode du gradient qui peut toutefois affecter la convergence de l'algorithme.

Modèle cinématique inverse

Question 9

D'abord, le Modèle Cinématique Inverse (MCI) est fait par le fichier `MCI.m`, on a décrit les consignes comme de points où l'organe terminale auraient arrivé s'il suivit parfaitement la trajectoire entre les poins avec la vitesse spécifié et capté dans les moments kTe où $k \in N$. En plus, lorsqu'on utilise la fonction `MGI`, on envoie toujours le même valeurs de k_{max}, $alpha_{max}$ et $epsilon_x$.

Bien que la trajectoire demandée est rectiligne, la trajectoire suivie est n'en est pas comme on voit dans la Figure 5 grâce à des erreurs (Figure 6). On note aussi dans la Figure 5 cette erreur parce que la position désirée finale (X_{df}) est loin de la position du organe terminal à la fin de l'exécution (O_{Ef}).

Question 10

L'évolution temporelle des variables q_1 à q_6 est faite par le fichier `Q09Test.m` aussi. La Figure 7 représente l'évolution temporelle pour chaque variable articulaire séparément. Remarquez que la coordonnée articulaire 5 sort de la zone allouée deux fois et ces moments sont exactement desquels l'erreur accroît.

Question 11

En faisant les changements nécessaires pour prendre en compte le l'éloignement des bornes articulaires par les variables articulaires nous avons développé une nouvelle fonction

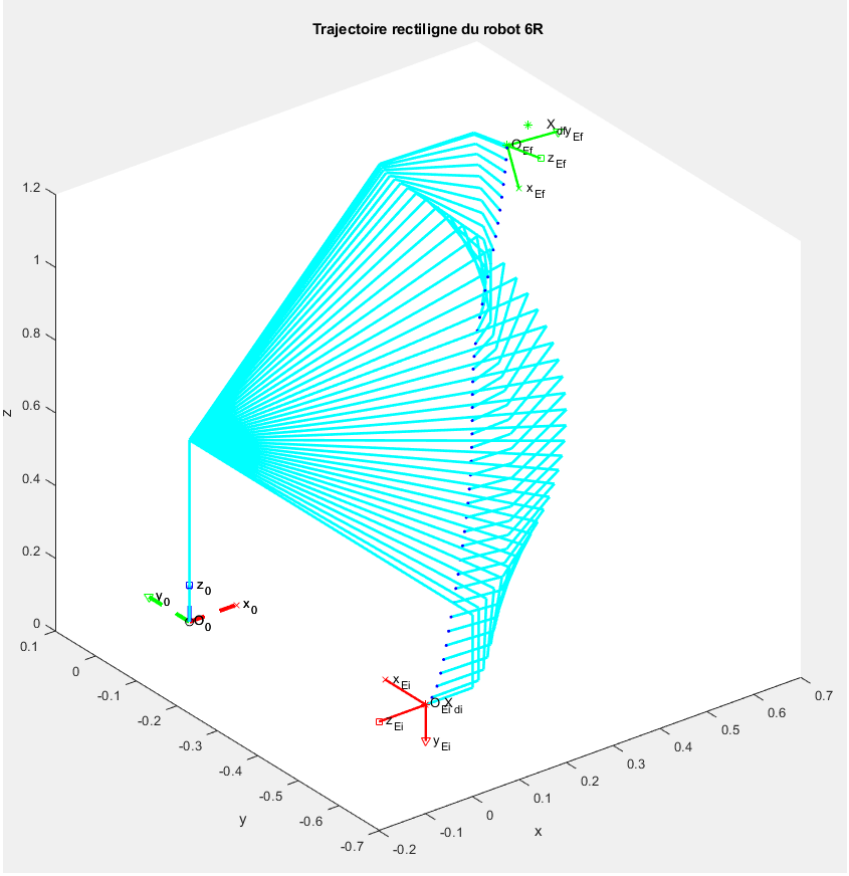

FIGURE 5: Trajectoire suivie par l'organe terminal en utilisant la Méthode Cinématique Inverse, c'est-à-dire la Méthode Géométrique Inverse pour chaque point de la trajectoire à chaque pas de temps.

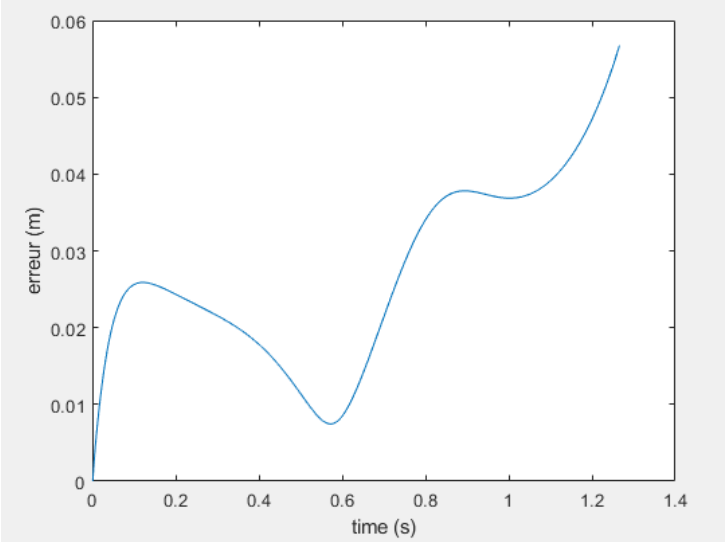

FIGURE 6: Erreur au cours du temps de la trajectoire suivie avec la méthode MCI.

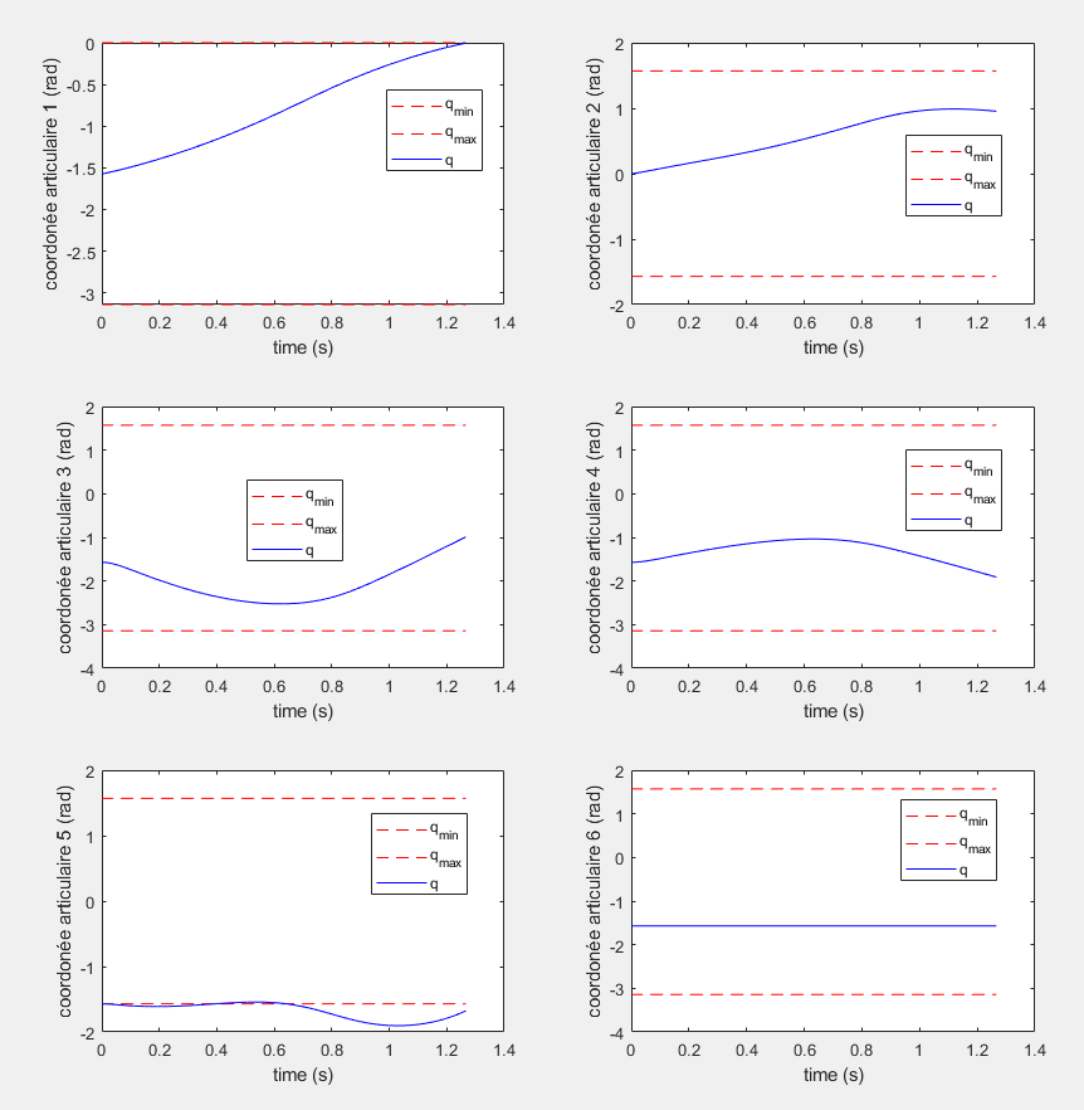

FIGURE 7: Variables articulaires du robot manipulateur au cours d'une trajectoire rectiligne en utilisant la méthode MCI.

présente sur le fichier `MCIbutees.m` qui considère une tâche secondaire visant l'éloignement des butées articulaires q_{min} et q_{max}.

Figure 10 représente l'évolution temporelle de chaque variable articulaire séparément et la Figure 9 affiche l'erreur de suivie de trajectoire de l'organe terminal.

On note que contrairement à ce qu'on voit dans la question 10, les coordonnées articulaires sont toujours dedans l'intervalle de q_{min} à q_{max}, la trajectoire suivie semble rectiligne et l'erreur est si petit qu'on ne voit pas une différence importante entre la coordonnée opéra-

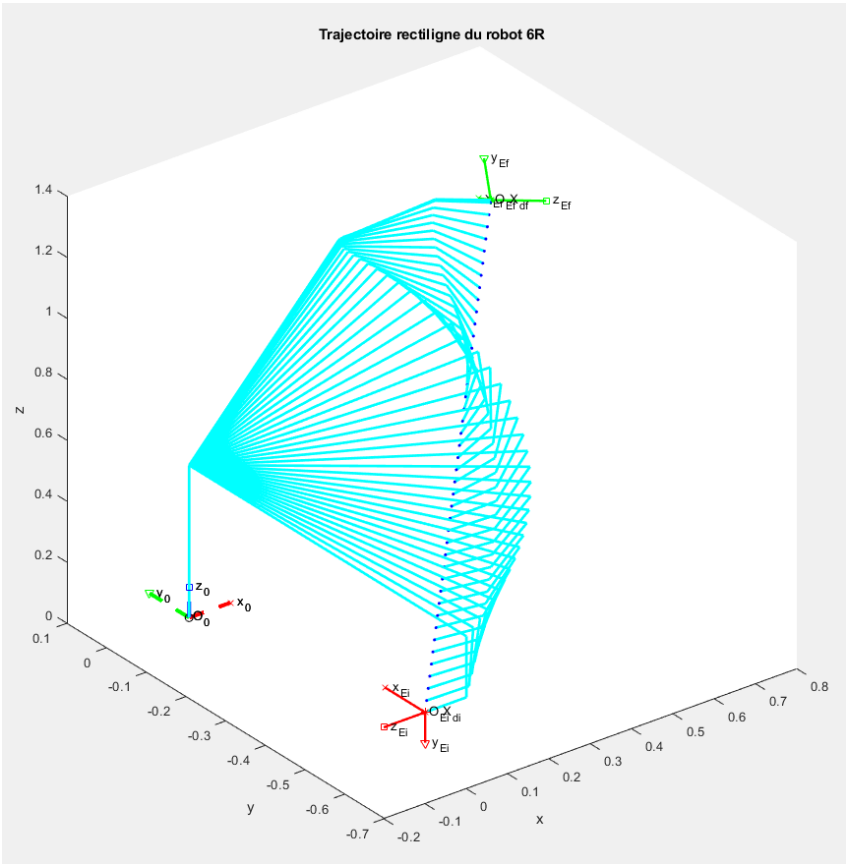

FIGURE 8: Trajectoire suivie par l'organe terminal en utilisant la Méthode Cinématique Inverse avec éloignement des bornes articulaires..

tionnelle désirée (X_{df}) et la position du organe terminale à la fin (O_{Ef})

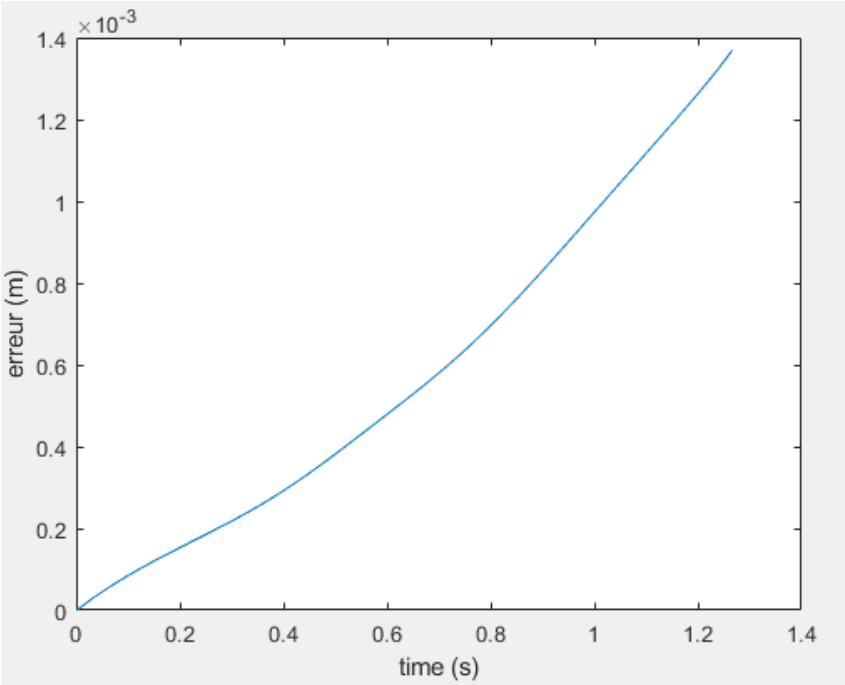

FIGURE 9: Erreur au cours du temps de la trajectoire suivie avec la méthode MCI avec éloignement des bornes articulaires.

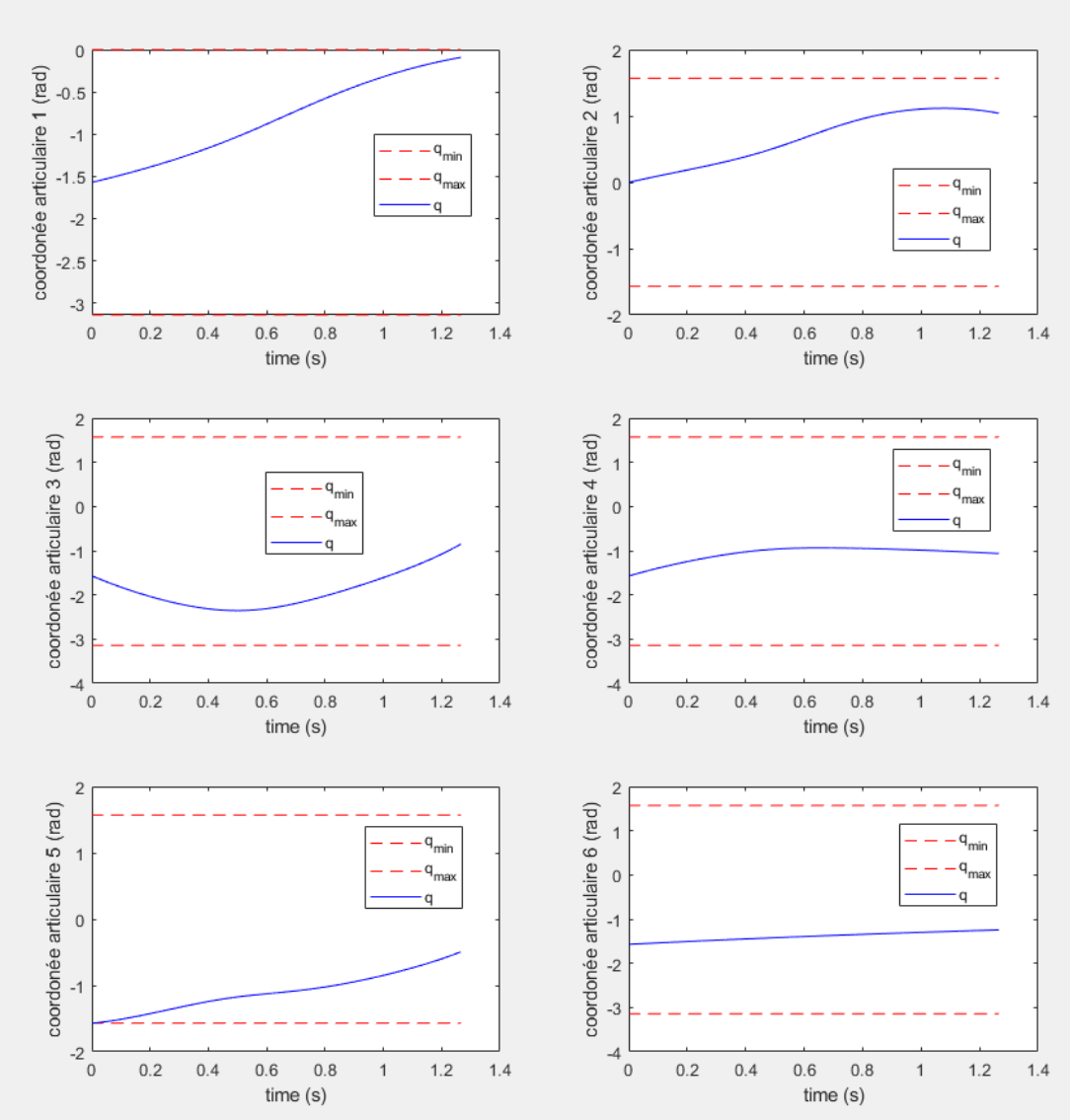

FIGURE 10: Variables articulaires du robot manipulateur au cours d'une trajectoire rectiligne en utilisant la méthode MCI avec éloignement des bornes articulaires.

TP n°2 - Dynamique et Commande

Modèle Dynamique

Question 12

On a développé la fonction exigée dans le fichier `CalculMatriceJacobienneGi`. Pour essayer cette fonction, tournez `Q12Test.m` qui fait l'appel à cette fonction et détermine la vitesse V_{Gi} du centre de masse G_i et la vitesse de rotation ω_i de tous les corps Ci dans le repère R_0 pour un vecteur de coordonnées articulaires donné.

Question 13

On a développé la fonction qui a pour but retourner la matrice d'inertie du robot pour une configuration. Pour l'éprouver, exécutez le fichier `Q13Test.m` pour un vecteur de coordonnées articulaires donné.

Question 14

Pour cette question, la fonction `CalculBornesMatriceInertie` a été développé pour calculer les bornes inférieures et supérieures à partir des valeurs propres de la matrice d'inertie. L'approche est de faire une recherche stochastique pour ces bornes afin d'éviter le maximal ou minimum locales, ensuite on utilise le gradient pour trouver le maximum et minimum local qui sera probablement globales aussi et pour assurer un résultat dedans une tolérance. Pour l'exécuter, il suffit de lancer le fichier `Q14Test.m`.

Les résultats pour les paramètres $k_{stoq} = 10000$ (numéro max d'itérations de la recherche stochastique), $k_{grad} = 1000$ (numéro max d'itérations de la recherche pour gradient), $alpha_{step} = 3$ (taux d'apprentissage initial), $tol = 0.000001$ (tolérance) et 10 testes sont montrés ci-dessous.

μ_1 = 0.013269 en moyenne avec une écart de $1,3787\ 10^{-6}$ μ_2 = 11,2897 en moyenne avec une écart de $7,54142\ 10^{-6}$

Ces petits écarts de l'ordre de la tolérance montre la robustesse et précision de notre algorithme.

Question 15

En exécutant le fichier `Q15Test.m`,vous pouvez tester la fonction développé `CalculCoupleGravite` qui lors d'un appel retourne le vecteur de couples articulaires de gravité du robot.

Question 16

L'algorithme développé pour trouver le borne du vecteur de couples de articulaires de gravité est analogue au algorithme de la Question 14 sauf qu'on ne calcule que le borne supérieure et que on utilise la norme de $G(q)$ au lieu des valeurs singulières de la matrice d'inertie. En exécutant le fichier `Q16Test.m`, on obtient pour les mêmes paramètres que l'autre question les résultats suivants :

g_b = 134,8646 en moyenne avec une écart de $0,025822$

Cette fois, l'algorithme n'atteint pas à la tolérance, mais l'ordre de l'écart est déjà très petite.

Question 17

Vous pouvez accéder la fonction et le modèle *Simulink* exigés dans les fichiers `CalculCouple Frottement.m` et `Modele_dynamique_direct.slx` respectivement. Il y a encore le fichier de teste aussi `Q17Test.m`, il faut juste choisir les module à tester et régler les paramètres.

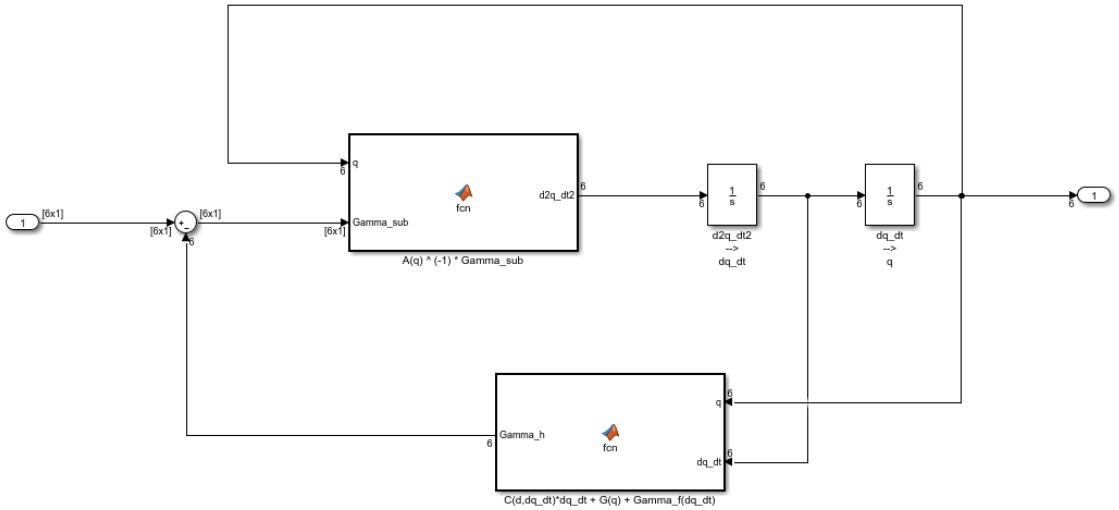

FIGURE 11: Modèle dynamique direct développé sous *Simulink*.

Génération de Trajectoire Articulaire

Question 18

D'abord, on a calculé le vecteur de accélérations maximales :

$$k_a = [44,2882; 44,2882; 44,2882; 31,0018; 31,0018; 31,0018]$$

Ensuite, on a découverte le temps total pour chaque articulation :

$$t_f = [0,3611; 0,3611; 0,3611; 0,4315; 0,4315; 0,4315]$$

Ainsi, le temps global et final de la trajectoire doit être 0,4315 s.

En lançant le fichier `Q17.m`, on regarde la trajectoire à suivre produite par notre fonction de planification hors ligne. De plus, on remarque que la fonction créée pour retourner les consignes peut être trouver dans le fichier `GenerationTrajP5HorsLigne.m`.

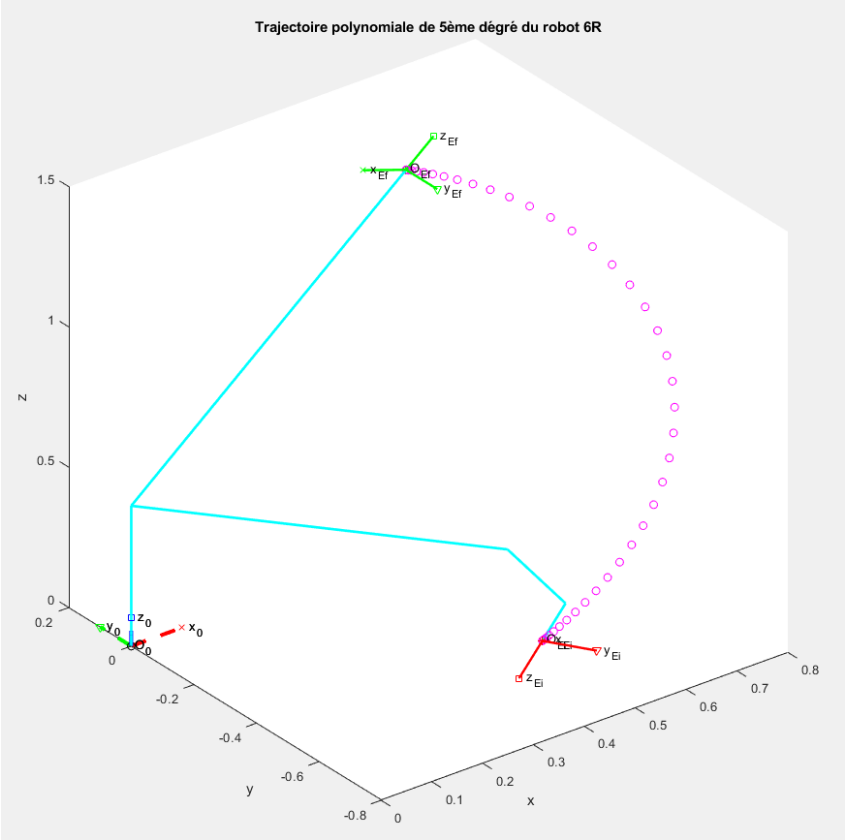

FIGURE 12: Trajectoire à suivre par l'organe terminal de forme polynomial d'ordre 5.

Question 19

On a développé le modèle *Simulink* `GeneTraj.slx` avec un sélecteur de chaînes de bus et un fonction GeneTraj qui réalise ce qu'on veut dans le fichier `GeneTraj.m`(Figure 13). Pour regarder la trajectoire articulaire à suivre, il faut juste ouvrir lancer le fichier `Q19Test` et ouvrir l'image montrée par le module `scope` du modèle modèle *Simulink*.

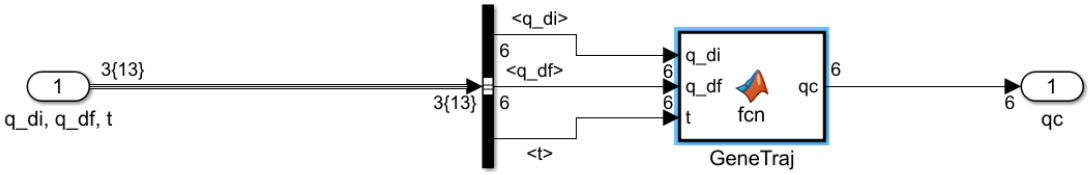

FIGURE 13: Modèle de génération de trajectoire articulaire développé sous *Simulink*.

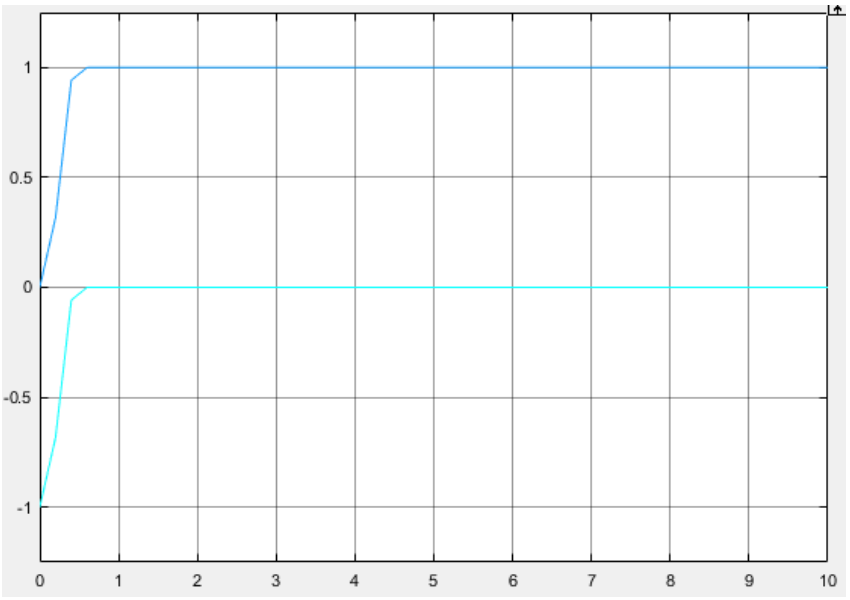

FIGURE 14: Trajectoire à suivre par l'organe terminal de forme polynomial d'ordre 5 sous *Simulink*.

On note que en effet les articulations vont de -1 à 0 ou 0 à 1 vers 0,5 s d'accord avec l'anticipation.

Commande dans l'Espace Articulaire

Question 20

La loi de commande a été implémenté dans le bloc de la Figure 15. On remarque qu'on utilise un bloc de dérivation dans le commande, ça n'est pas recommandé car il peut augmenter le bruit et nuire l'action de contrôle.

La simulation a été réalisé pour 10 s en tournant le fichier `Q20Test.m` avec des gains articulaires proportionnel K_d et dérivative égales à les matrices ci-dessous.

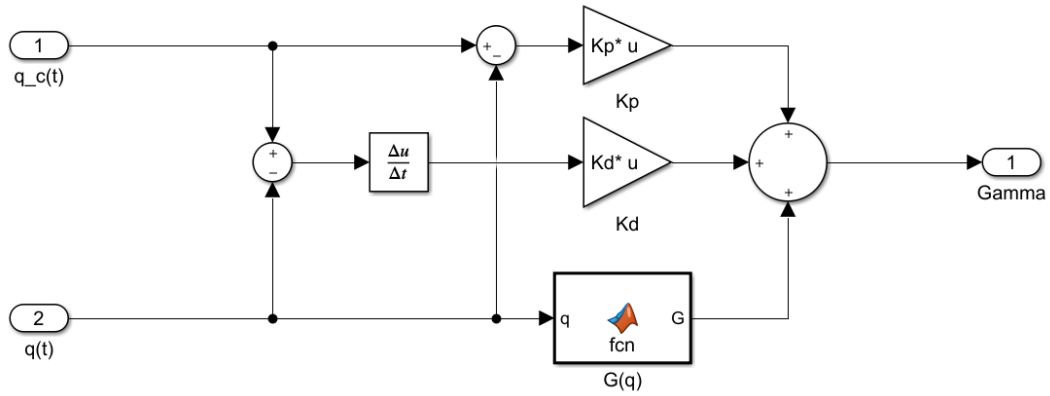

FIGURE 15: Commande P.D. décentralisé avec compensation de la gravité sous *Simulink*.

$$K_P = \begin{bmatrix} 59.6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 38.22 & 0 & 0 & 0 & 0 \\ 0 & 0 & 204.4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 190 & 0 & 0 \\ 0 & 0 & 0 & 0 & 190 & 0 \\ 0 & 0 & 0 & 0 & 0 & 190 \end{bmatrix}$$

$$K_P = \begin{bmatrix} 176 & 0 & 0 & 0 & 0 & 0 \\ 0 & 157.56 & 0 & 0 & 0 & 0 \\ 0 & 0 & 246.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 227.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 227.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 227.5 \end{bmatrix}$$

En regardant les coordonnées articulaires (Figure 16) et leur erreur (Figure 17) au cours du temps, on aperçoit que les coordonnées articulaires se stabilisent dans moins de 10 s, surpassent les consignes au début avec un erreur plus important et qu'on réussit à réduire

l'erreur des coordonnées articulaires du poignet au plus de 0,05 rad cependant l'erreur des coordonnées articulaires du porteur se sont révélées plus compliqués à régler. Ces erreurs sont évidents dans la Figure 18 car la différence non négligeable entre les consignes et les positions réelles du organe terminale.

Concernant les efforts, les couples articulaires sont toujours dedans l'intervalle permise comme on voit dans la Figure 19. L'articulation plus exigée est l'articulation 2 qui est laquelle avec le plus grand erreur.

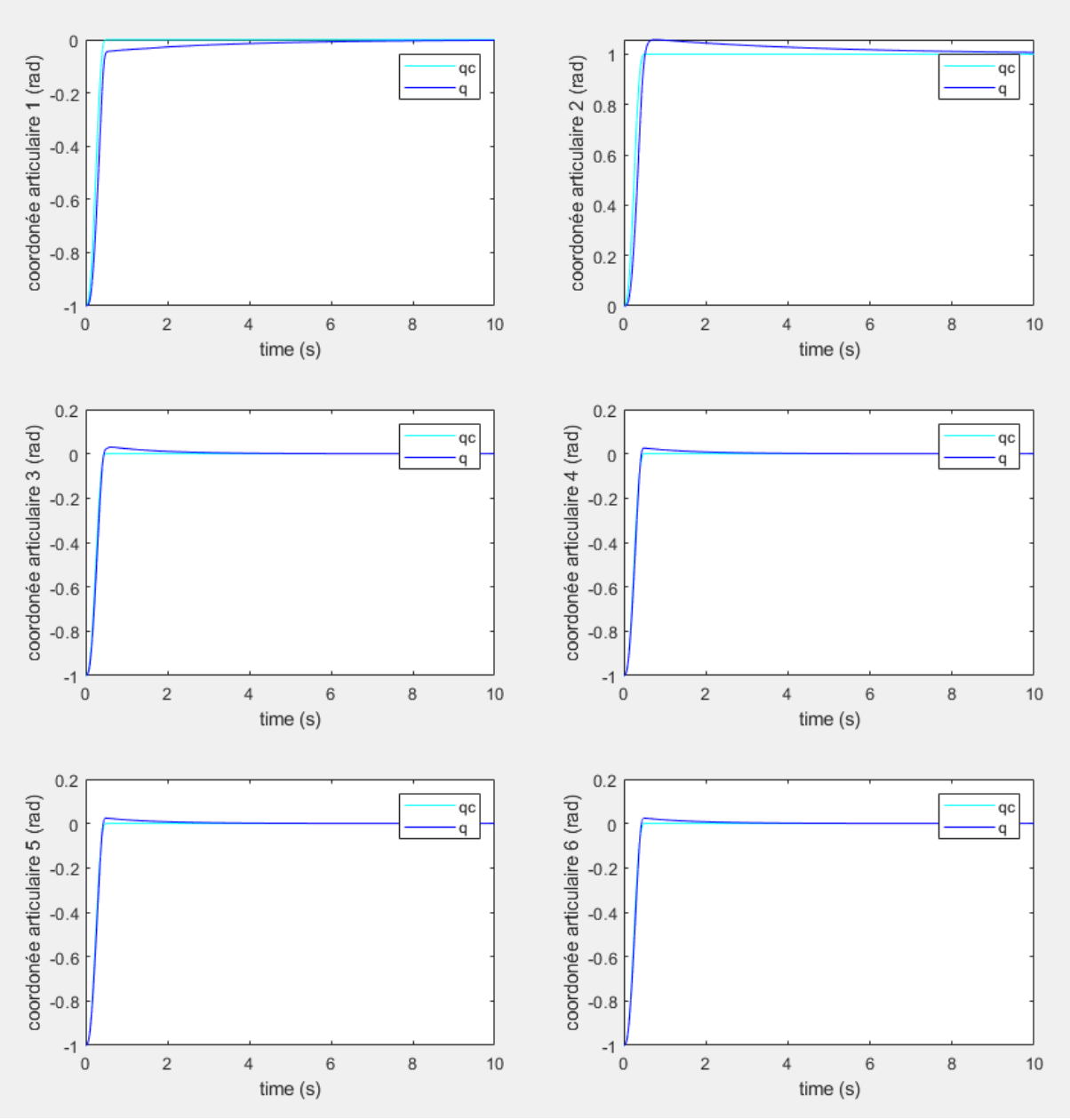

FIGURE 16: Coordonnées articulaires consignes et réelles au cours du temps avec un commande P.D. décentralisé avec compensation de la gravité.

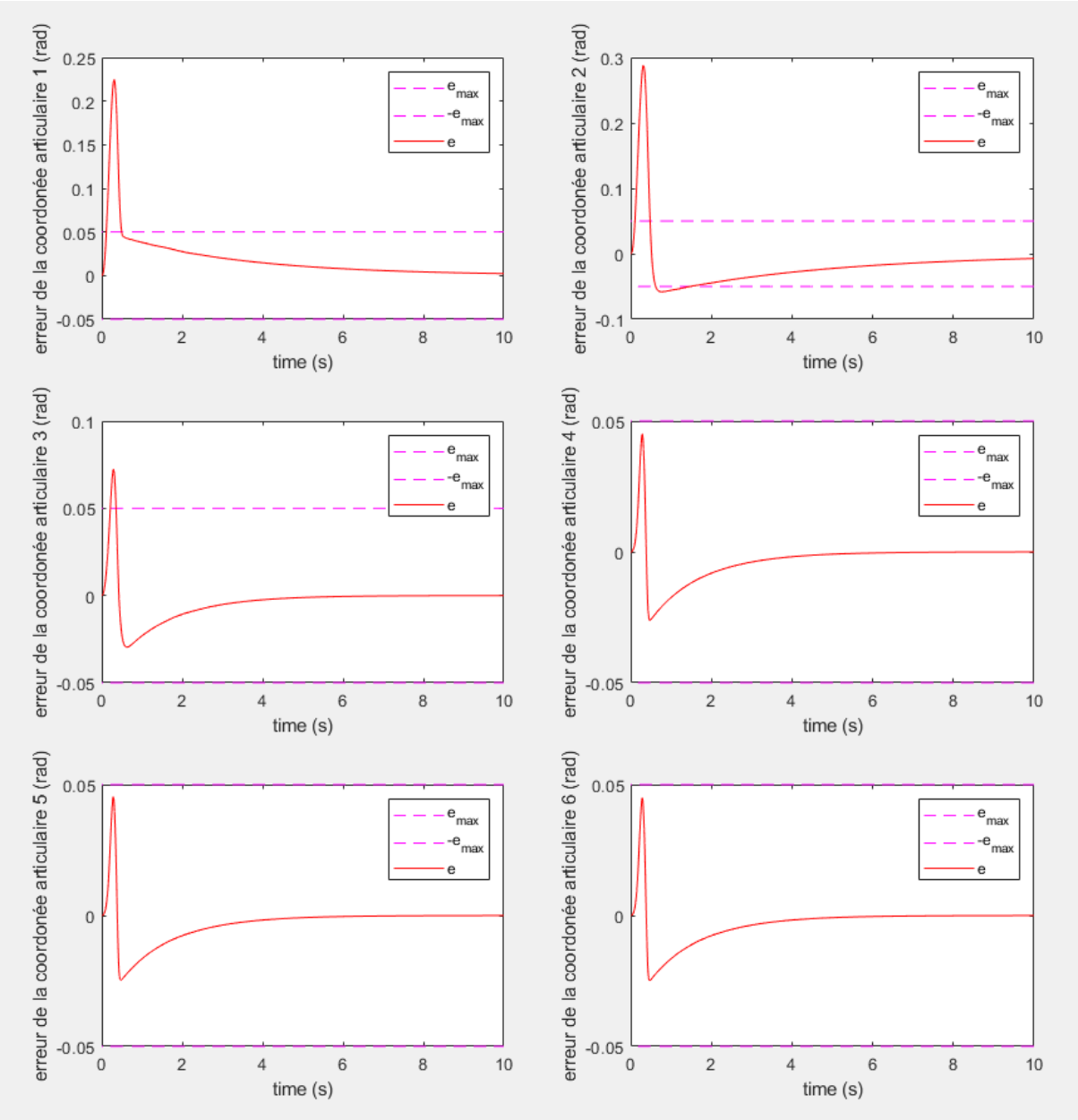

FIGURE 17: Erreur de chaque coordonnée articulaire au cours du temps avec un commande P.D. décentralisé avec compensation de la gravité.

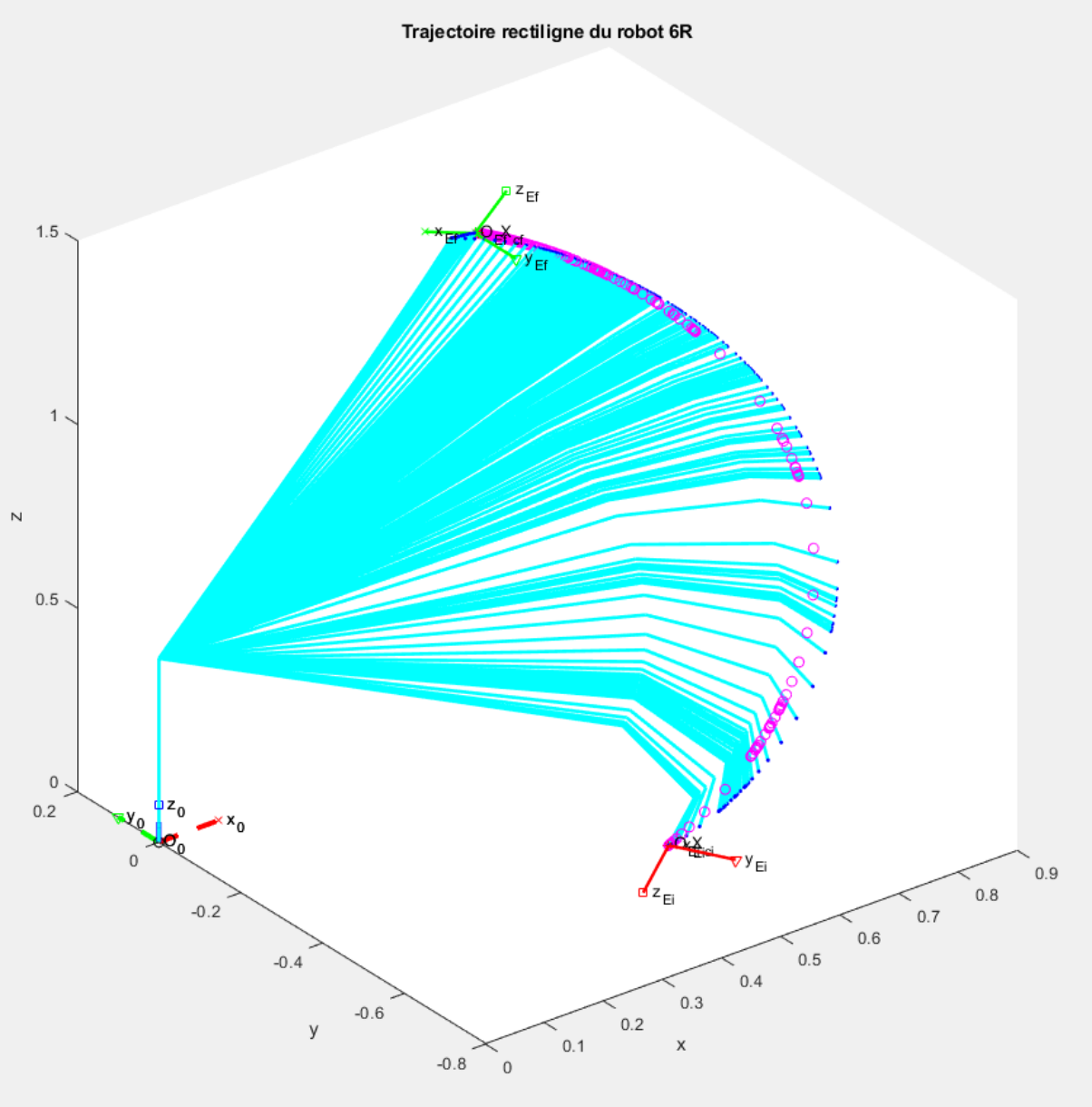

FIGURE 18: Trajectoire à suivre (en magenta) et suivie (en bleue) par l'organe terminal avec un commande P.D. décentralisé avec compensation de la gravité.

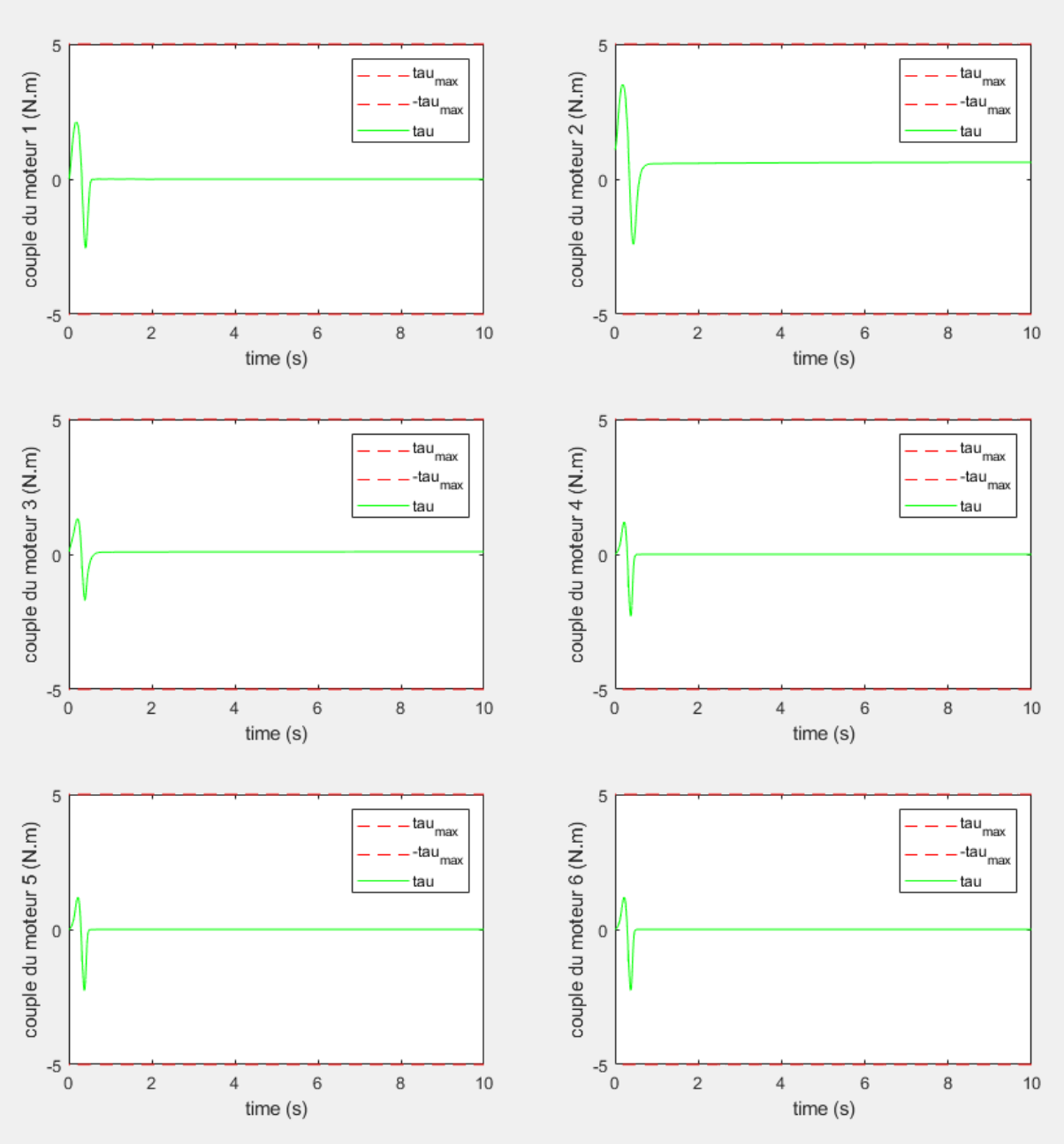

FIGURE 19: Couples articulaires de commande vus entre le moteur et la réduction au cours du temps avec un commande P.D. décentralisé avec compensation de la gravité.