UPSSITECH

Rapport de projet

Bureau d'études
Bras manipulateur RRPR

Auteurs	Constant ROUX Bernard PARFAITE Simon BERLUNGA Peter PIRIOU–DEZY

Promotion	SRI 2024 - Année 2022
Destinataire	Michel TAÏX
Date	Décembre 2022

Contents

1 Modélisation du robot

1.1 Placement des repères et paramètres de DHM

Nous commençons par placer les repères nécessaires pour la modélisation (Figure 1) de façon à annuler le plus possible de paramètres de Denavit-Hartenberg (Table 1). De plus, nous ajoutons un repère associé à l'outil pour faciliter la suite.

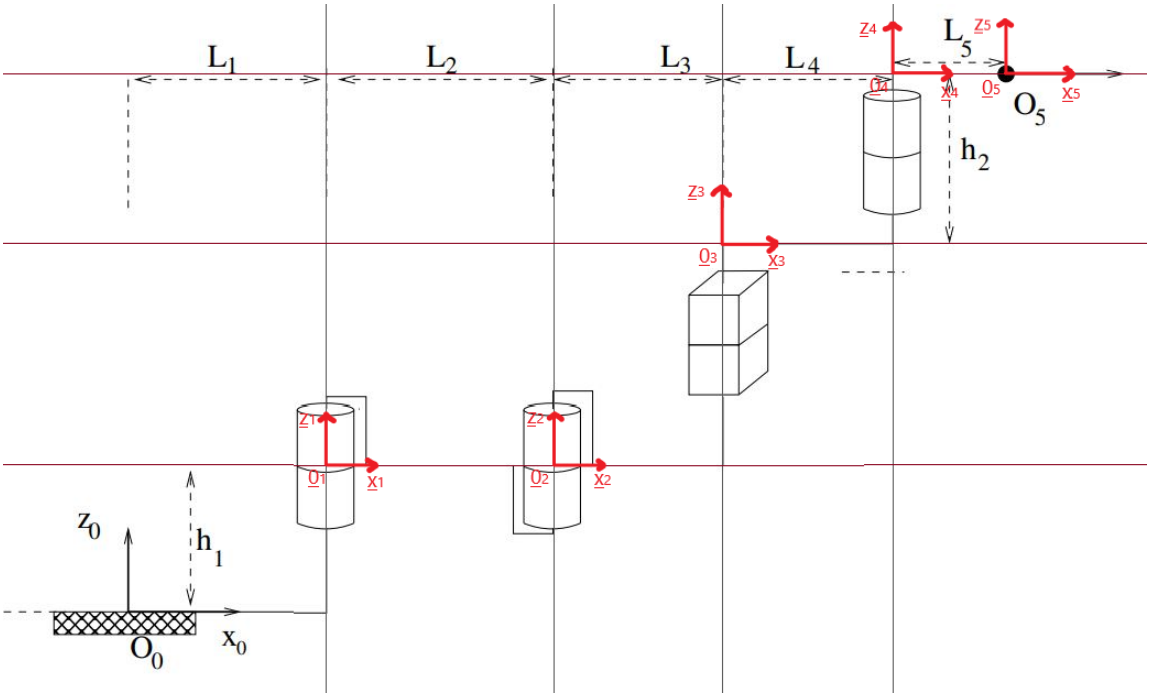

Figure 1: Repères de Denavit-Hartenberg dans la configuration figure du robot RRPR.

Table 1: Paramètres de Denavit-Hartenberg pour un robot RRPR.

L_i	1	2	3	4
σ_i	0	0	1	0
a_{i-1}	L_1	L_2	L_3	L_4
α_{i-1}	0	0	0	0
r_i	h_1	0	q_3	h_2
θ_i	q_1	q_2	0	q_4
q_{iFIG}	0	0	> 0	0

1.2 Modèle géométrique direct

À l'aide des paramètres calculés précédemment, nous calculons les quatre matrices de transformation en fonction des configurations des liaisons (Matrices 1, 2, 3, 4) ainsi que la matrice de

transformation constante vers le repère outil (Matrice 5).

$$T_{01}(q_1) = \begin{pmatrix} c_1 & -s_1 & 0 & L_1 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & h_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

$$T_{12}(q_2) = \begin{pmatrix} c_2 & -s_2 & 0 & L_2 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2}$$

$$T_{23}(q_3) = \begin{pmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

$$T_{34}(q_4) = \begin{pmatrix} c_4 & -s_4 & 0 & L_4 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 1 & h_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

$$T_{45} = \begin{pmatrix} 1 & 0 & 0 & L_5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

Nous en déduisons ainsi la matrice de transformation de $\Re_0$ à $\Re_5$ (Équation 6, non développée ici car calculée numériquement).

$$T_{05} = T_{01} \cdot T_{12} \cdot T_{23} \cdot T_{34} \cdot T_{45} \tag{6}$$

1.3 Modèle géométrique indirect

Nous employons la méthode de Paul avec $p = 2$ (Équation 7) pour résoudre le modèle géométrique inverse en prenant directement comme matrice désirée la matrice $T_{05}(X, Y, Z, \theta)$ (Matrice 10) afin d'éviter une conversion par rapport à la méthode classique. En posant l'équation 8 avec les matrices 9 et 11, nous en déduisons l'équation 12 .

$$p = \frac{n}{2} \tag{7}$$

$$T_{20}(q_1, q_2) \cdot T_{05}(\underline{X}^*) = T_{25}(q_3, q_4) \tag{8}$$

$$T_{20}(q_1, q_2) = \begin{pmatrix} c_{1+2} & s_{1+2} & 0 & -L_1 c_{1+2} - L_2 c_2 \\ -s_{1+2} & c_{1+2} & 0 & L_1 s_{1+2} + L_2 s_2 \\ 0 & 0 & 1 & -h_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{9}$$

$$T_{05}(\underline{X}^*) = \begin{pmatrix} c_\theta & -s_\theta & 0 & X \\ s_\theta & c_\theta & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{10}$$

$$T_{25}(q_3, q_4) = \begin{pmatrix} c_4 & -s_4 & 0 & L_3 + L_4 + c_4 L_5 \\ s_4 & c_4 & 0 & s_4 L_5 \\ 0 & 0 & 1 & q_3 + h_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{11}$$

$$T_{20}(q_1, q_2) T_{05}(\underline{X}^*) = \begin{pmatrix} c_\theta c_{1+2} + s_\theta s_{1+2} & c_\theta s_{1+2} - s_\theta c_{1+2} & 0 & -L_1 c_{1+2} - L_2 c_2 + X c_{1+2} + Y s_{1+2} \\ s_\theta c_{1+2} - c_\theta s_{1+2} & c_\theta c_{1+2} + s_\theta s_{1+2} & 0 & L_1 s_{1+2} + L_2 s_2 - X s_{1+2} + Y c_{1+2} \\ 0 & 0 & 1 & Z - h_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{12}$$

Par identification avec l'équation 12, nous posons le système 13 qui peut être résolu en le mettant sous la forme d'équations dont les solutions sont connues (Système résolu 14).

$$\begin{cases} s_\theta c_{1+2} - c_\theta s_{1+2} = s_4 \\ c_\theta c_{1+2} + s_\theta s_{1+2} = c_4 \\ -L_1 c_{1+2} - L_2 c_2 + X c_{1+2} + Y s_{1+2} = L_3 + L_4 + c_4 L_5 \\ L_1 s_{1+2} + L_2 s_2 - X s_{1+2} + Y c_{1+2} = s_4 L_5 \\ Z - h_1 = q_3 + h_2 \\ \theta = q_1 + q_2 + q_4 \end{cases} \tag{13}$$

$$\begin{cases} q_3 = Z - h_1 - h_2 \\ q_2 = arctan_2(\epsilon\sqrt{1 - c_2^2}, c_2) \\ q_1 = arctan_2(s_{1+2}, c_{1+2}) - q_2 \\ q_4 = \theta - q_1 - q_2 \end{cases} \tag{14}$$

La résolution de q_2 possède deux solutions, ce qui implique que l'arbre des solutions possède deux branches en fonction de ϵ (Figure 2).

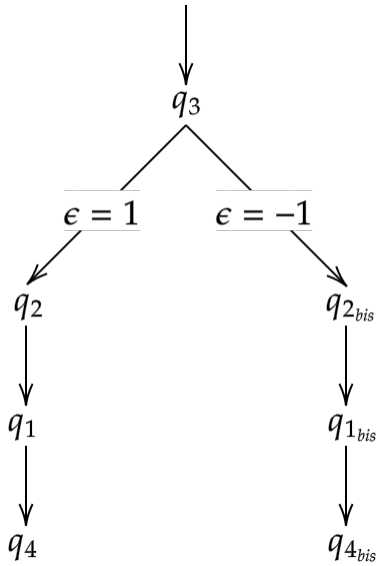

Figure 2: Arbre des solutions du MGI

Finalement, il existe une singularité lorsque $s_{1+2} = c_{1+2} = 0$ ($atan2(0,0)$ est indéterminé). Il faudra donc prendre en compte ce cas lors du développement du modèle.

1.4 Modèle différentiel direct

La différence entre la jacobienne analytique et géométrique réside dans les vitesses angulaires calculées. En effet, la jacobienne analytique représentera les vitesses angulaires de l'orientation de l'organe terminal tel qu'elle a été modélisée, là où la jacobienne géométrique calculera les vitesses angulaires autour des axes x, y et z. Dans notre cas, il est plus intéressant de calculer la jacobienne analytique, car l'orientation de notre organe terminal est directement fonction du paramètre θ (Matrice 15).

$$J(q) = \begin{pmatrix} -L_2.s_1 - L_3.c_2.s_1 - L_3.s_2.c_1 & -L_3.c_2.s_1 - L_3.s_2.c_1 & 0 & 0 \\ -L_4.c_2.s_1 - L_4.s_2.c_1 & -L_4.s_1.c_2 - L_4.s_2.c_1 & & \\ L_2.c_1 + L_3.c_2.c_1 - L_3.s_2.s_1 & L_3.c_2.c_1 - L_3.s_2.s_1 & 0 & 0 \\ +L_4.c_2.c_1 - L_4.s_2.s_1 & +L_4.c_2.c_1 - L_4.s_2.s_1 & & \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \quad (15)$$

Nous remarquons que deux lignes sont entièrement nulles, nous pouvons donc simplifier la jacobienne analytique (Matrice 16). Elle a alors la dimension d'une matrice carrée, ce qui va rendre possible le calcul inverse de cette dernière (dans le cas où son déterminant n'est pas nul).

$$J(q) = \begin{pmatrix} -L_2.s_1 - L_3.c_2.s_1 - L_3.s_2.c_1 & -L_3.c_2.s_1 - L_3.s_2.c_1 & 0 & 0 \\ -L_4.c_2.s_1 - L_4.s_2.c_1 & -L_4.s_1.c_2 - L_4.s_2.c_1 & & \\ L_2.c_1 + L_3.c_2.c_1 - L_3.s_2.s_1 & L_3.c_2.c_1 - L_3.s_2.s_1 & 0 & 0 \\ +L_4.c_2.c_1 - L_4.s_2.s_1 & +L_4.c_2.c_1 - L_4.s_2.s_1 & & \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \quad (16)$$

1.5 Modèle différentiel indirect

Le modèle différentiel indirect, étant l'inverse du modèle différentiel direct, est calculé numériquement en inversant la jacobienne analytique $J(q)$. Il faut simplement vérifier que le déterminant de $J(q)$ n'est pas nul pour que cette dernière soit inversible.

2 Génération de mouvements

Avant d'expliciter la génération de mouvements (segment ici), nous allons introduire les variables nécessaires à cette partie :

- $A = \begin{pmatrix} x_A & y_A & z_A \end{pmatrix}$: les coordonnées géométriques du point de départ de la trajectoire ;
- $B = \begin{pmatrix} x_B & y_B & z_B \end{pmatrix}$: les coordonnées géométriques du point d'arrivée de la trajectoire ;
- θ : l'angle désiré de l'organe terminal sur le plan (xy) ;
- d : la distance euclidienne entre les points A et B ;
- $U = \begin{pmatrix} x_U & y_U & z_U \end{pmatrix} = \begin{pmatrix} x_B - x_A & y_B - y_A & z_B - z_A \end{pmatrix}$: le vecteur représentant les distances entre chaque composante des points A et B ;
- V : la vitesse maximale atteinte par loi de mouvement avec un profil de vitesse triangulaire symétrique ;
- t_1 : le temps de commutation à la moitié de la loi de mouvement avec un profil de vitesse triangulaire symétrique ;
- t_2 : le temps de commutation de fin de la loi de mouvement avec un profil de vitesse triangulaire symétrique.

2.1 Loi de mouvement

Le cahier des charges nous impose une loi de commande en vitesse avec un profil de triangle symétrique (Équation 17). En dérivant cette dernière, nous trouvons alors la loi de commande en accélération correspondante (Équation 19) et en l'intégrant la loi de position correspondante (Équation 18).

$$\dot{s}(t) = \begin{cases} \frac{V}{t_1} \cdot t & \text{si } 0 \leq t \leq t_1 \\ -\frac{V}{t_1} \cdot t + 2 \cdot V & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ 0 & \text{sinon} \end{cases} \tag{17}$$

$$s(t) = \begin{cases} \frac{V}{2 \cdot t_1} \cdot t^2 & \text{si } 0 \leq t \leq t_1 \\ -\frac{V}{2 \cdot t_1} \cdot t^2 + 2 \cdot V \cdot t - d & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ 0 & \text{sinon} \end{cases} \tag{18}$$

$$\ddot{s}(t) = \begin{cases} \frac{V}{t_1} & \text{si } 0 \leq t \leq t_1 \\ -\frac{V}{t_1} & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ 0 & \text{sinon} \end{cases} \tag{19}$$

2.2 Temps de commutation

La loi étant symétrique, nous savons qu'à l'instant t_1, nous avons parcouru la moitié de la distance d. En se basant sur la première équation de la loi de commande en position, nous pouvons déduire t_1 (Équation 20). Toujours par symétrie, l'instant t_2 vaut deux fois l'instant t_1 (Équation 21).

$$\frac{V}{2 \cdot t_1} \cdot t_1^2 = \frac{d}{2} \iff t_1 = \frac{d}{V} \tag{20}$$

$$t_2 = 2 \cdot t_1 \tag{21}$$

2.3 Trajectoire géométrique

La loi de commande en position correspond à une abscisse curviligne s. Nous souhaitons que la trajectoire géométrique aille du point A au point B en suivant le vecteur U de la droite (AB) et en évoluant de 0% ($\frac{0}{d}$) à 100% ($\frac{s_{final}}{d}$) comme présenté dans l'équation 22.

$$X(s) = \begin{pmatrix} x_A + U_x \cdot \frac{s}{d} \\ y_A + U_y \cdot \frac{s}{d} \\ z_A + U_z \cdot \frac{s}{d} \end{pmatrix} \tag{22}$$

2.4 Trajectoire temporelle

Afin d'obtenir la trajectoire temporelle pour chaque axe, il suffit de remplacer s dans l'équation 22 par son expression (Équation 18). Nous obtenir ainsi les expressions 23, 24, 25, 26 pour chaque composante, non développées ici, car implémentées de manière numérique.

$$X(s(t)) = \begin{pmatrix} x(s(t)) \\ y(s(t)) \\ z(s(t)) \end{pmatrix} \tag{23}$$

$$x(t) = \begin{cases} x_A + \frac{U_x}{d} \cdot \frac{V}{2 \cdot t_1} \cdot t^2 & \text{si } 0 \leq t \leq t_1 \\ x_A + \frac{U_x}{d} \cdot (-\frac{V}{2 \cdot t_1} \cdot t^2 + 2 \cdot V \cdot t - d) & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ x_A & \text{sinon} \end{cases} \tag{24}$$

$$y(t) = \begin{cases} y_A + \frac{U_y}{d} \cdot \frac{V}{2 \cdot t_1} \cdot t^2 & \text{si } 0 \leq t \leq t_1 \\ y_A + \frac{U_y}{d} \cdot (-\frac{V}{2 \cdot t_1} \cdot t^2 + 2 \cdot V \cdot t - d) & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ y_A & \text{sinon} \end{cases} \tag{25}$$

$$z(t) = \begin{cases} z_A + \frac{U_z}{d} \cdot \frac{V}{2 \cdot t_1} \cdot t^2 & \text{si } 0 \leq t \leq t_1 \\ z_A + \frac{U_z}{d} \cdot (-\frac{V}{2 \cdot t_1} \cdot t^2 + 2 \cdot V \cdot t - d) & \text{si } t_1 \leq t \leq 2 \cdot t_1 \\ z_A & \text{sinon} \end{cases} \tag{26}$$

3 Primitive de mouvement

3.1 Algorithme

Afin de générer la trajectoire dans l'espace des configurations, il faut suivre les étapes suivantes :

- Vérifier si la trajectoire est réalisable par le robot ;
- Calculer les points de la trajectoire (position et vitesse) dans l'espace opérationnel en fonction d'une loi de commande ;
- Déterminer les configurations et les dérivées des configurations pour chaque point de la trajectoire précédente à l'aide des modèles géométrique et cinématique inverses.

Ces étapes sont résumées dans l'algorithme suivant écrit en pseudo-code :

Algorithm 1 $traj(A, B, V, \theta)$

if is_not_reachable(A, B, θ) **then**
    raise NotReachableError
end if
t, M, $\dot{M} \leftarrow get\_M(A, B, V)$
q, $q_{bis} \leftarrow$ MGI.get_q(M, θ)
$\dot{q} \leftarrow$ MDI.get_$\dot{q}(q, \dot{M}, \theta)$
$\dot{q}_{bis} \leftarrow$ MDI.get_$\dot{q}(q_{bis}, \dot{M}, \theta)$
return t, q, q_{bis}, $\dot{q}$, $\dot{q}_{bis}$

3.2 Zone atteignable

Sur le plan (xy), l'aire de la zone atteignable par l'outil peut se traduire par des équations de cercles. Nous pouvons déduire aisément que le point O_5 sur ce plan se situe sur le disque creux de rayon intérieur $L_2 - L_3 - L_4$, de rayon extérieur $L_2 + L_3 + L_4$ et de centre $\begin{pmatrix} L_1 + L_5 \cdot cos(\theta) & L_5 \cdot sin(\theta) \end{pmatrix}$ suivant l'équation 27 et visible sur la figure 3.

$$(L_2 - L_3 - L_4)^2 \leq (x - L_1 - L_5 \cdot cos(\theta))^2 + (y - L_5 \cdot sin(\theta))^2 \leq (L_2 + L_3 + L_4)^2 \qquad (27)$$

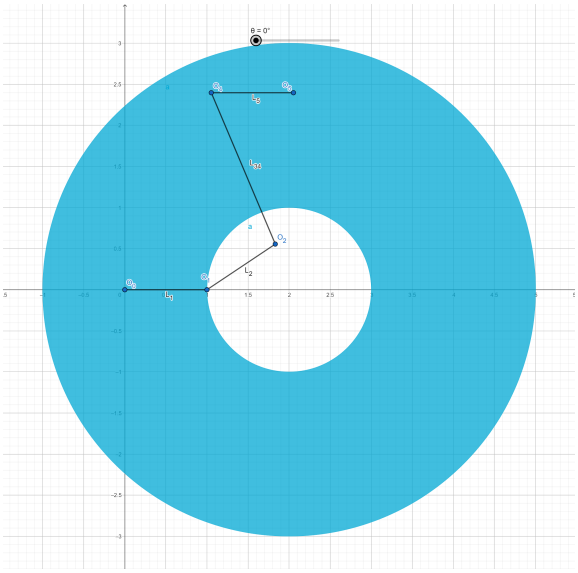

Figure 3: Visualisation de l'aire atteignable par le robot (en bleu) sur le plan (xy) avec $\theta = 0$ rad, $L_1 = L_2 = L_3 = L_4 = L_5 = 1$.

Nous pouvons maximiser l'aire de la zone atteignable en réduisant le rayon intérieur à zéro. Cela se traduit par l'équation 28 et la nouvelle aire est visible sur la figure 4.

$$L_2^* = L_3 + L_4 \tag{28}$$

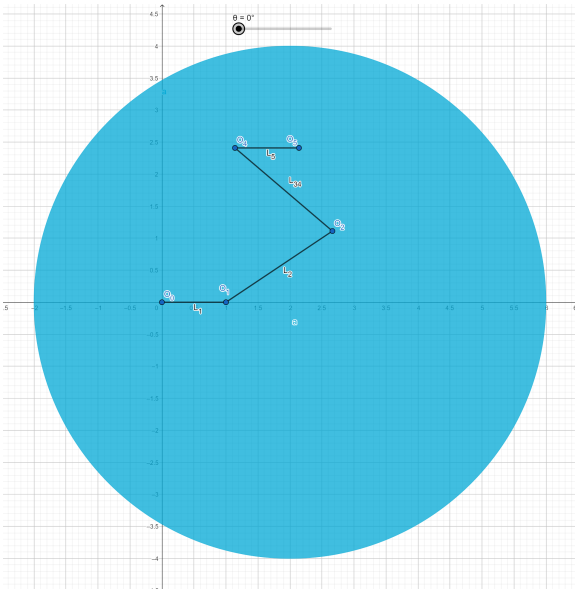

Figure 4: Visualisation de l'aire atteignable par le robot (en bleu) sur le plan (xy) avec $\theta = 0$ rad, $L_2 = L_3 + L_4$, $L_1 = L_3 = L_4 = L_5 = 1$.

Dans cette configuration, il suffit alors de vérifier que les points A et B appartiennent au cercle de rayon $2 \cdot (L_3 + L_4)$ et de centre $\begin{pmatrix} L_1 + L_5 \cdot cos(\theta) & L_5 \cdot sin(\theta) \end{pmatrix}$ pour savoir que le robot pourra atteindre tous les points du segment $[AB]$.

4 Fonctionnalités supplémentaires

4.1 Visualisation du MGD

Afin de simplifier la visualisation du MGD et la vérification du MGI, nous avons mis en place un visualiseur 3D du robot en fonction de la configuration choisie via des curseurs (Figure 5).

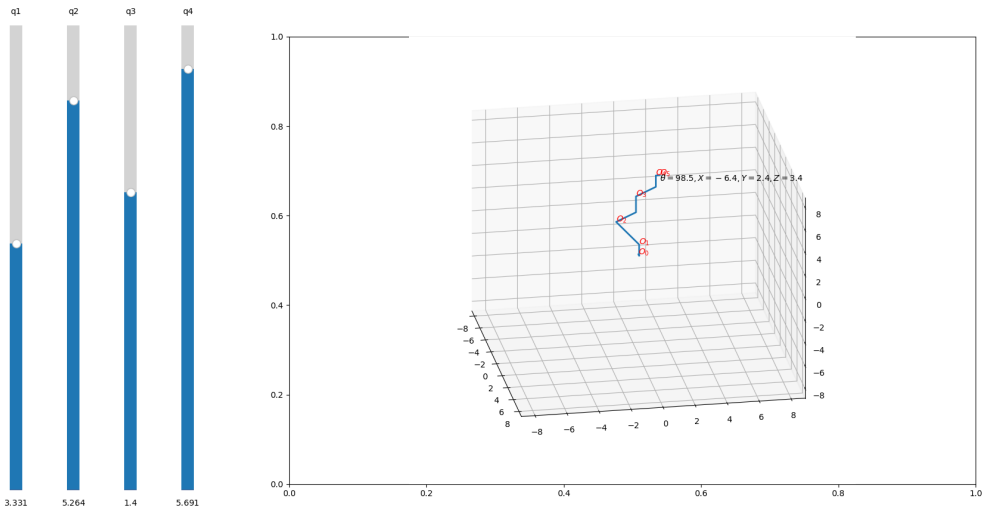

Figure 5: Visualiseur 3D du MGD du robot RRPR.

4.2 Visualisation du résultat de la fonction *traj*

En se basant sur le même code que le visualiseur 3D du MGD, nous avons mis en place un visualiseur 3D du résultat de la fonction *traj*. Celui-ci consiste en un curseur permettant de visualiser les deux configurations du robot calculées en fonction de l'itération $k.T_e$. De plus, la courbe géométrique 3D attendue y est superposée pour pouvoir vérifier la cohérence des résultats (Figure 6).

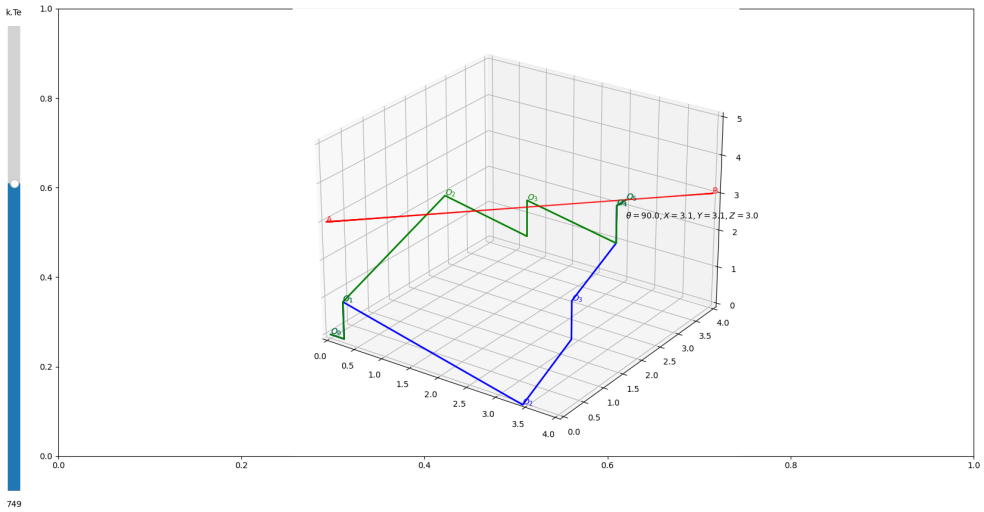

Figure 6: Exemple de visualisation avec le visualiseur 3D du résultat de la fonction *traj* du robot RRPR.

4.3 Trajectoire d'arcs de cercle

Le code étant très modulaire, il a été aisé d'ajouter une nouvelle trajectoire géométrique, celle des arcs de cercle sur le plan (xy). La fonction *traj* a cette fois pour paramètres A le point de départ, B le point d'arrivée, C le centre du cercle, V la vitesse maximale du profil de la loi de commande, *clockwise* le sens de rotation (horaire ou anti-horaire) et θ l'orientation de l'organe terminal (Voir exemple figure 7). Avec α_A, α_B, α, d et r respectivement $\widehat{\overrightarrow{x_0}, \overrightarrow{A}}$, $\widehat{\overrightarrow{x_0}, \overrightarrow{B}}$, $\alpha_B - \alpha_A[2\pi]$, la longueur de l'arc et le rayon de l'arc, nous pouvons écrire l'équation géométrique du cercle en fonction de l'abscisse curviligne (Équation 29). La vérification permettant de savoir si l'arc est atteignable ou non par le robot n'a pas été implémentée.

$$X(s) = \begin{pmatrix} x_C + r \cdot cos(\pm\alpha \cdot \frac{s}{d} + \alpha_A) \\ y_C + r \cdot sin(\pm\alpha \cdot \frac{s}{d} + \alpha_A) \\ z_A \end{pmatrix} \tag{29}$$

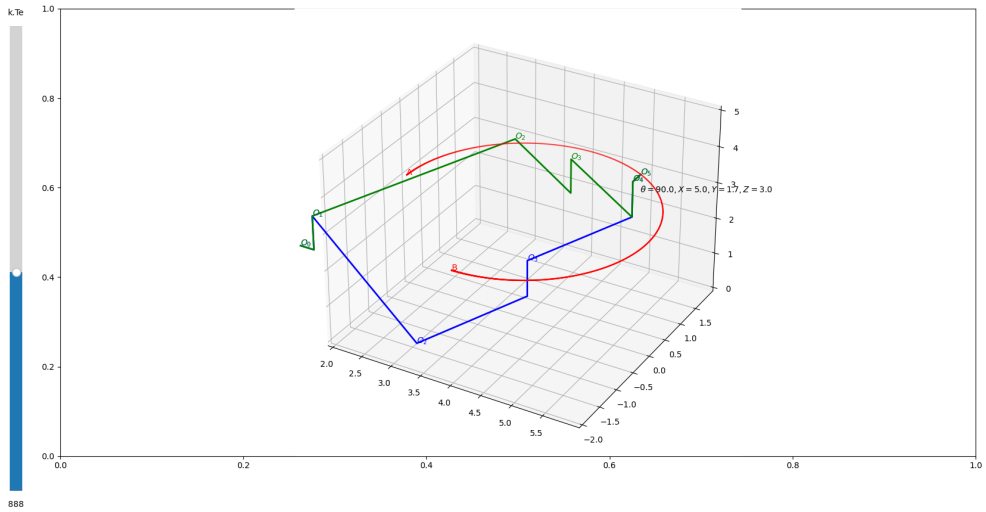

Figure 7: Exemple de visualisation avec le visualiseur 3D du résultat de la fonction *traj* du robot RRPR pour un arc de cercle.

4.4 Interpréteur et visualiseur de commandes G-Code

Possédant à présent le moyen de générer des segments dans l'espace et des arcs de cercle dans le plan, il est tout à fait possible d'interpréter les commandes G-code $G0$ et $G1$ pour les segments et $G2$ et $G3$ pour les arcs de cercle. Une fois interprétées, nous pouvons les mettre au format de nos fonctions pour déduire les trajectoires opérationnelles et donc aussi les trajectoires dans l'espace des configurations (Figures 8 et 9).

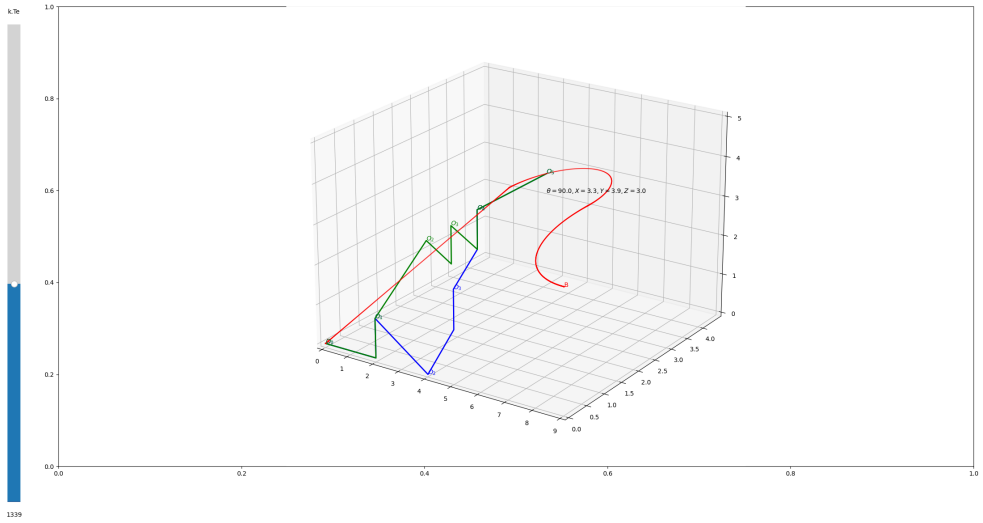

Figure 8: Exemple de visualisation avec le visualiseur 3D du résultat de la fonction *traj* du robot RRPR pour la succession de 3 commandes G-codes différentes.

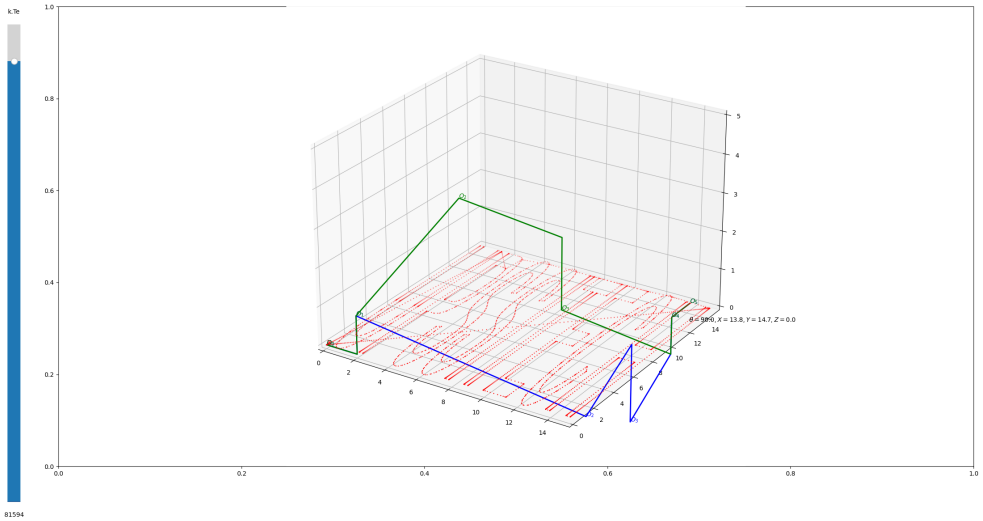

Figure 9: Exemple de visualisation avec le visualiseur 3D du résultat de la fonction *traj* du robot RRPR pour la succession de + de 300 commandes G-codes.

5 Exemples

5.1 Exemple 1

Nous allons d'abord présenter un exemple fonctionnel avec tous les résultats associés et obtenables depuis le logiciel fourni. Cet exemple montre le fonctionnement normal du robot avec une trajectoire de droite dans l'espace. Les paramètres fournis par l'utilisateur sont les suivants :

- $A = \begin{pmatrix} 1 & 0 & 3 \end{pmatrix}$;
- $B = \begin{pmatrix} 4 & 2 & 5 \end{pmatrix}$;
- $\theta = \frac{\pi}{2}$ rad ;
- $V = 1\ m.s^{-1}$;
- $T_e = 0.01$ s ;
- $H = \begin{pmatrix} 1 & 1 \end{pmatrix}$;
- $L = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 \end{pmatrix}$.

Nous générons d'abord la loi de commande en vitesse avec un profil en triangle symétrique. Cette dernière est correctement générée, car la vitesse atteint bien au maximum le V imposé, la loi en position tend vers d et $t_1 = \frac{d}{V} = d$ vaut bien d (Figure 10).

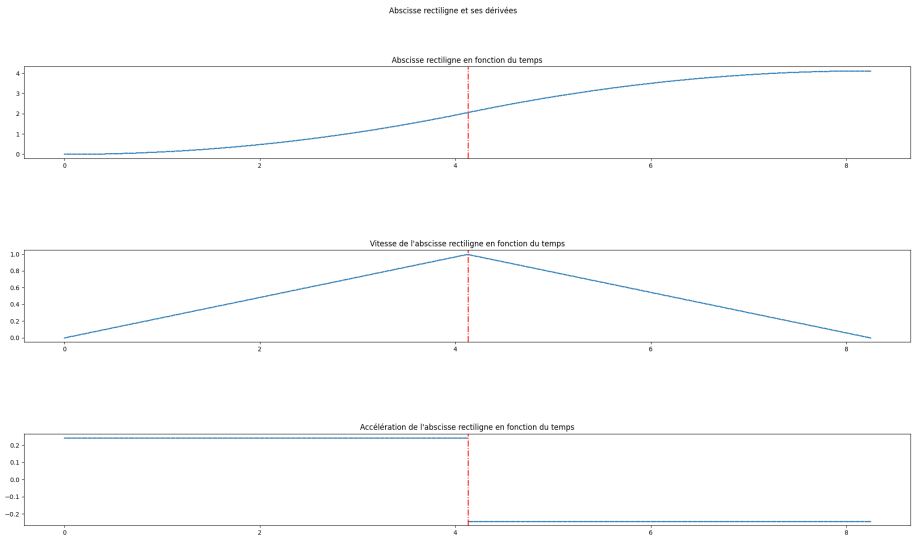

Figure 10: Abscisse curviligne et ses dérivées en fonction de $V = 1\ m.s^{-1}$ la vitesse maximale et $d = \sqrt{17}$ m la distance euclidienne entre les points A et B en bleu ; Instant t_1 en pointillé rouge.

En appliquant la loi de commande à la trajectoire en segment désirée, nous obtenons la figure 11 qui représente la trajectoire et ses dérivées effectuée pour chaque axe du repère cartésien. Nous constatons que la position de départ de chaque axe atteint bien la position d'arrivée en respectant la loi de commande en vitesse.

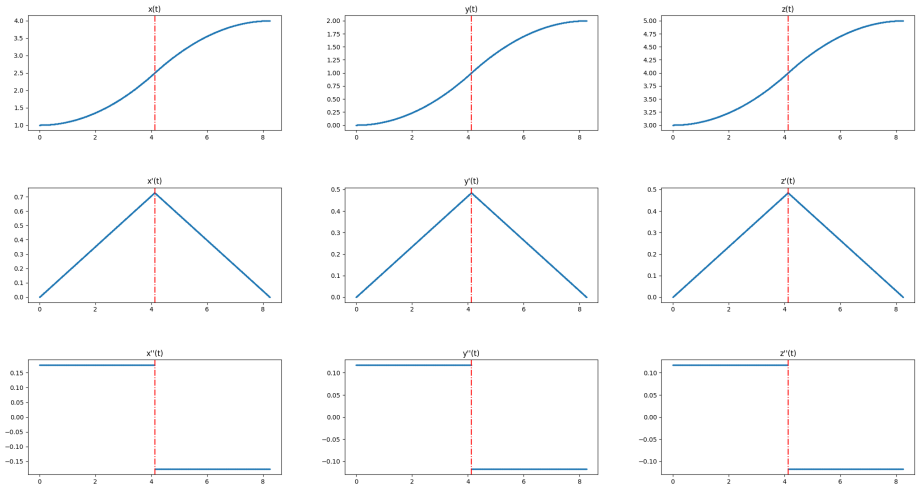

Figure 11: Trajectoire opérationnelle sur chaque composante en fonction du temps et ses dérivées en bleu ; Instant t_1 en pointillé rouge.

Nous pouvons aussi représenter θ en fonction du temps, qui, dans notre cahier des charges, est une simple constante. Il est envisageable de créer une nouvelle loi de commande exclusivement pour ce paramètre (Figure 12).

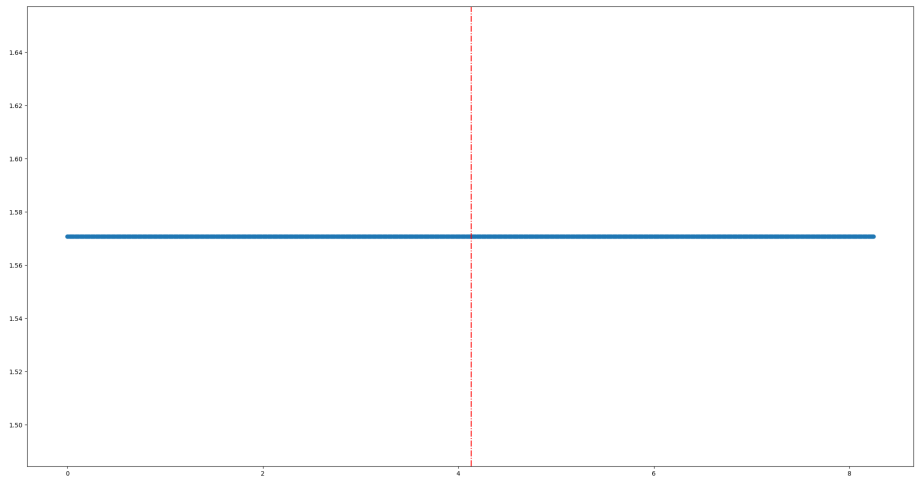

Figure 12: θ en fonction du temps en bleu ; Instant t_1 en pointillé rouge.

Finalement, nous représentons la trajectoire dans l'espace cartésien (Figure 13). L'axe bleu foncé représente l'orientation de l'organe terminal θ (animé sur le logiciel). Nous constatons qu'au début du segment les points échantillonnés sont de plus en plus écartés, ce qui se traduit par une augmentation de la vitesse. Cela prouve une fois de plus que la loi de commande est fonctionnelle.

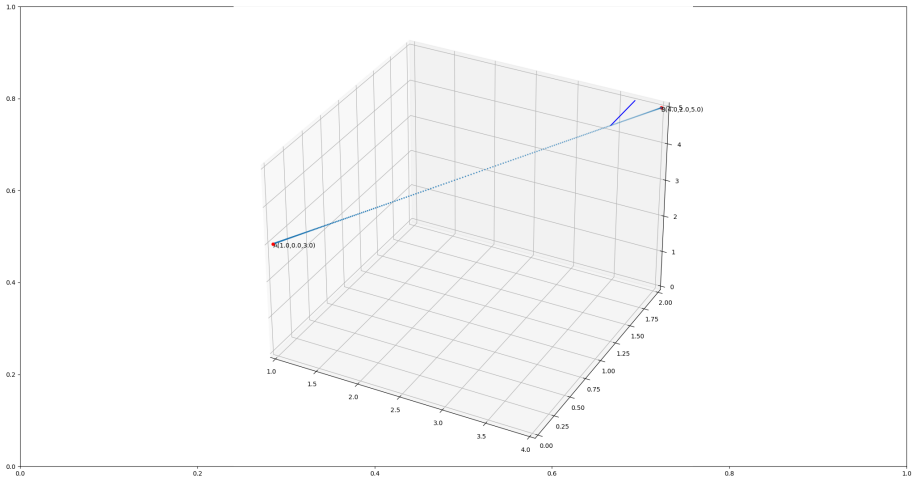

Figure 13: Trajectoire opérationnelle dans l'espace cartésien 3D en bleu ; Orientation de θ en bleu foncé.

En utilisant le MDI, nous obtenons pour chaque point de la trajectoire en vitesse deux configurations $\dot{q}$. Nous pouvons alors réinjecter les configurations obtenues dans les MDD afin de comparer les courbes de la vitesse attendue et les nouvelles courbes calculées. Sur la figure 14, nous constatons qu'elles sont identiques.

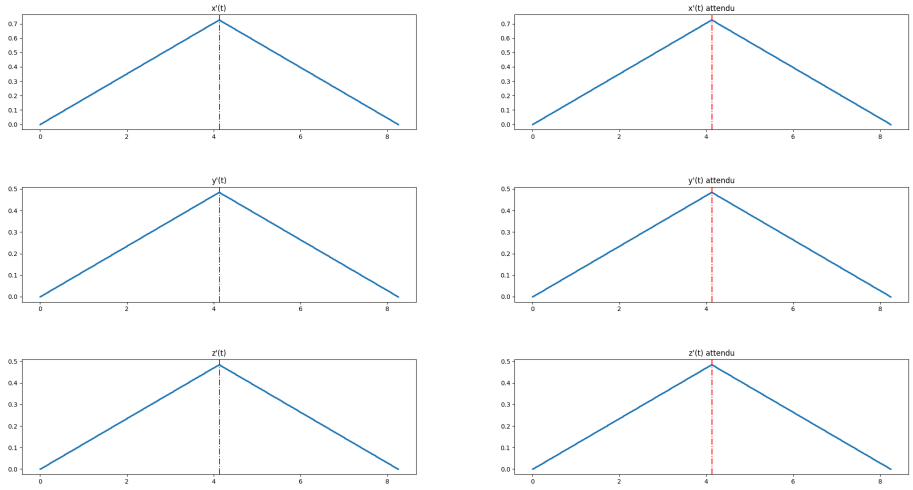

Figure 14: Vitesses des composantes du point O_5 calculées à l'aide du MDD à gauche ; Vitesses des composantes du point O_5 attendues.

À l'aide du MGI, nous pouvons ensuite calculer pour chaque point de la trajectoire les deux configurations associées. La représentation donnée ici semble discontinue (Figure 15) mais c'est parce que les configurations associées à des rotoïdes sont comprises entre 0 et 2π.

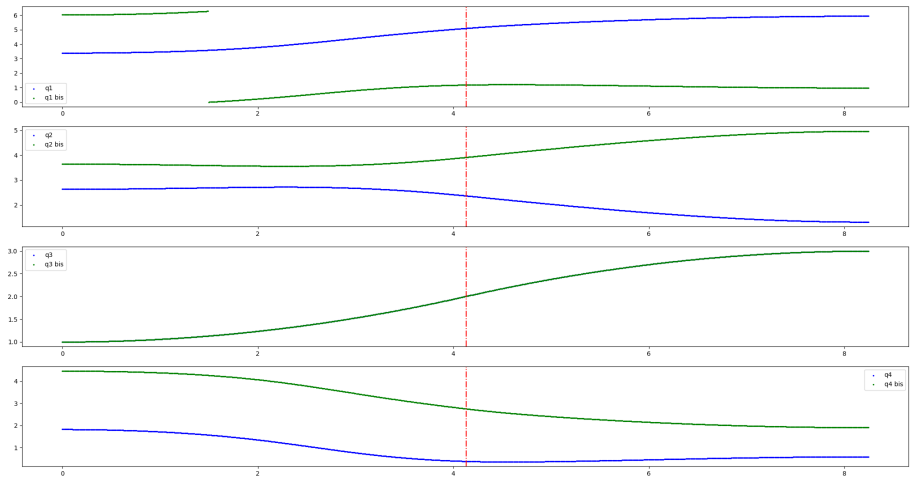

Figure 15: q_i en fonction du temps en bleu ; q_{ibis} en fonction du temps en vert ; Instant t_1 en pointillé rouge.

De la même manière, à l'aide du MDI, nous pouvons calculer pour chaque point de la trajectoire les deux dérivées de configurations associées (Figure 16).

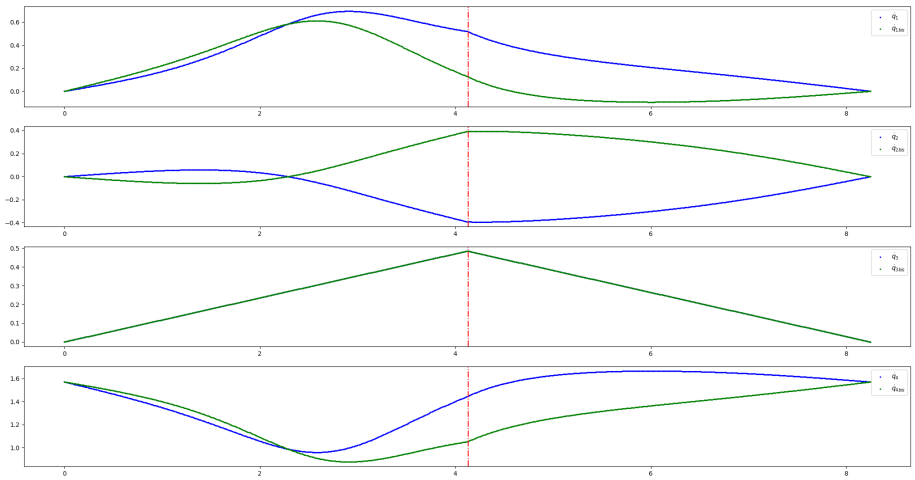

Figure 16: $\dot{q}_i$ en fonction du temps en bleu ; $\dot{q}_{ibis}$ en fonction du temps en vert ; Instant t_1 en pointillé rouge.

Finalement, nous pouvons simuler les mouvements de chaque axe du robot grâce aux configurations calculées pour vérifier si ce dernier suit bien la trajectoire désirée.

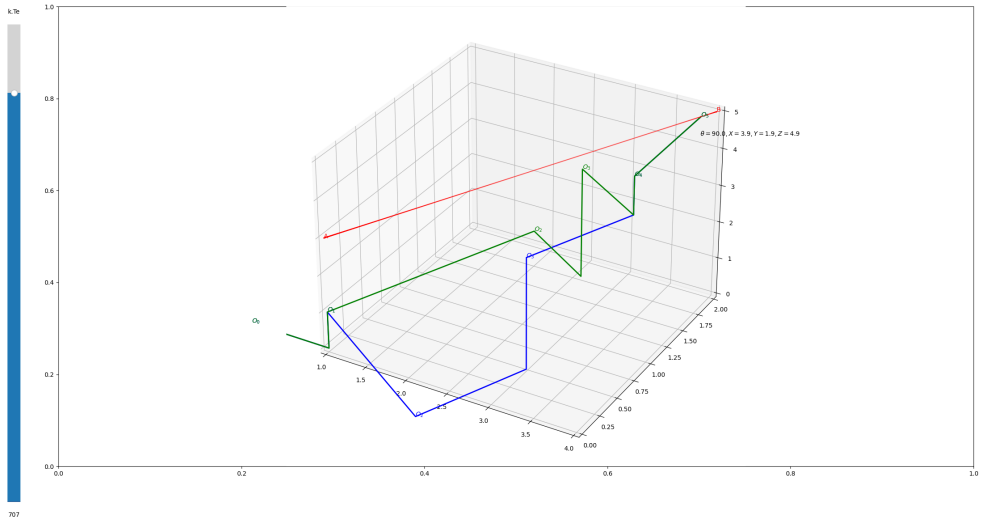

Figure 17: visualisation avec le visualiseur 3D du résultat de la fonction $traj$ du robot RRPR pour les paramètres utilisés.

5.2 Exemple 2

Nous allons maintenant présenter un exemple dans lequel un problème apparaît. Les paramètres fournis par l'utilisateur sont les suivants :

- $A = \begin{pmatrix} 0 & 0 & 3 \end{pmatrix}$;
- $B = \begin{pmatrix} 4 & 4 & 3 \end{pmatrix}$;
- $\theta = \frac{\pi}{2}$ rad ;
- $V = 1\ m.s^{-1}$;
- $T_e = 0.01$ s ;
- $H = \begin{pmatrix} 1 & 1 \end{pmatrix}$;
- $L = \begin{pmatrix} 1 & 4 & 2 & 2 & 1 \end{pmatrix}$.

Avec les paramètres donnés, nous voyons lors du calcul de q_1 un point erroné (Figure 18). Cela est dû à des arrondis de flottant proche d'une singularité. Cette erreur n'a pas encore pu être fixée par manque de temps.

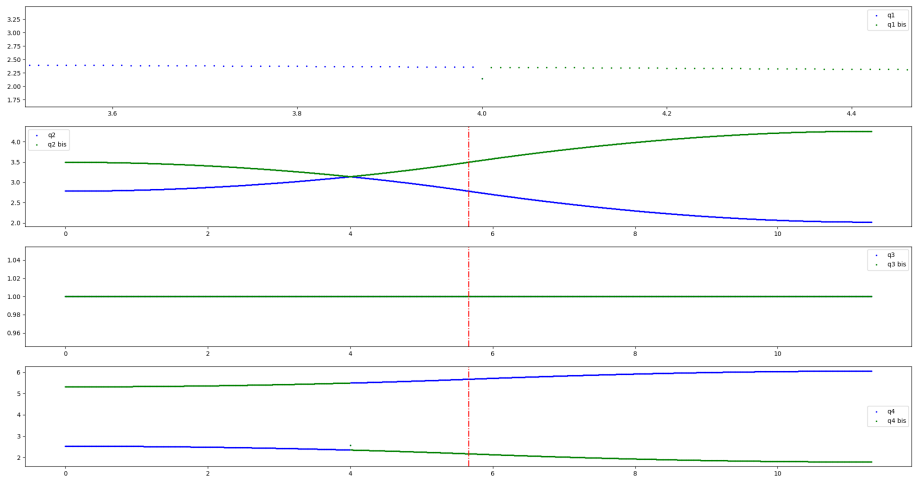

Figure 18: q_i en fonction du temps en bleu ; q_{ibis} en fonction du temps en vert ; Instant t_1 en pointillé rouge.

Cette erreur implique une très forte dérivée en un point, ce qui peut "emballer" le système (Figure 19). Il est possible de mettre simplement un cap pour éviter cela mais ça ne corrige pas l'erreur.

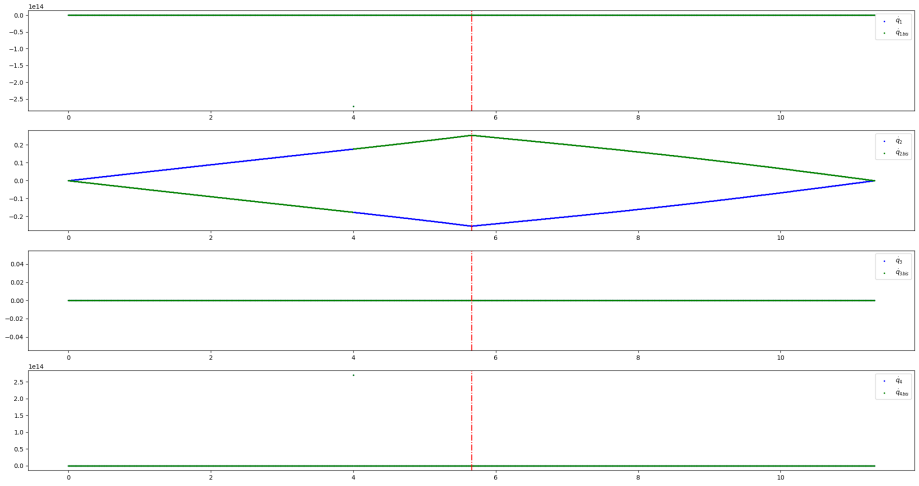

Figure 19: $\dot{q}_i$ en fonction du temps en bleu ; $\dot{q}_{ibis}$ en fonction du temps en vert ; Instant t_1 en pointillé rouge.