Elric Grandjean-Perrenoud-Comtesse
Adrian Gröger
Ciaran Van Hoeserlande

23 janvier 2022

Polytech Montpellier

# PROJET DE ROBOTIQUE DE MANIPULATION

ROBOTIS Manipulator-H, robot à six degrés de liberté

# SOMMAIRE

# INTRODUCTION

Dans le cadre de notre enseignement de spécialité du S7, nous avons dû proposer une façon de contrôler le ROBOTIS Manipulator-H, un robot à six degrés de liberté. Dans ce rapport, nous allons vous présenter comment nous nous y sommes pris pour réaliser ce projet.

L'intégralité du projet a été réalisée sur le logiciel Matlab.

Dans un premier temps, nous allons expliquer comment nous avons modélisé le robot sur le logiciel. Avec cela nous joindrons les différentes fonctions utilisées dans le cadre de cette partie.

Ensuite nous allons décrire les différentes méthodes de génération de mouvement du bras robotique. En premier lieu, nous décrirons le fonctionnement de la loi trapèze avec ses avantages et inconvénients. Puis nous ferons la description du fonctionnement de loi polynomiale de degré 5 avec, comme pour la loi trapèze, ses avantages et inconvénients.

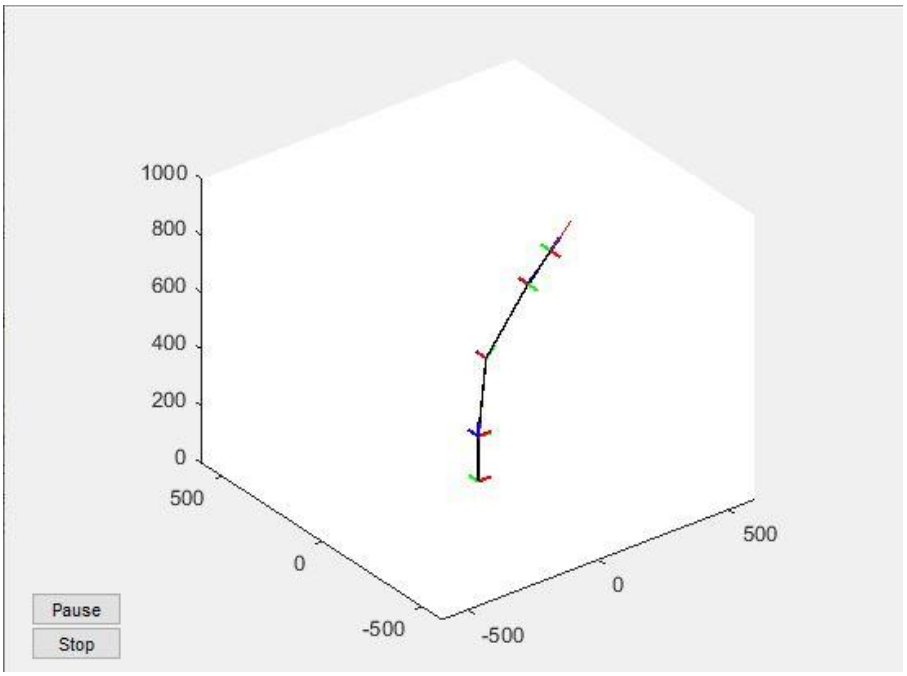
Enfin nous montrerons comment nous avons réalisé la commande cinématique du système avec une description du fonctionnement de notre code pour pouvoir la tester.

# MODÉLISATION DU ROBOT

Pour représenter le robot sur Matlab, nous avons utilisé les paramètres DH du robot avec tout de même un petit ajustement (précisé dans le sujet) afin d'avoir les mêmes dimensions que le modèle dans VREP.

| | $\alpha_j$ | $d_j$ | $\theta_j$ | $r_j$ |
|---|---|---|---|---|
| 1 | 0 | 0 | $\theta_1$ | 159 |
| 2 | $-\frac{\pi}{2}$ | 0 | $\theta_2 - \frac{\pi}{2} + tan^{-1}(\frac{30}{264}) = \theta_2 - 1.4576453$ | 0 |
| 3 | 0 | 265.69 | $\theta_3 - \frac{\pi}{4} - tan^{-1}(\frac{30}{264}) = \theta_3 - 0,898549163$ | 0 |
| 4 | $-\frac{\pi}{2}$ | 30 | $\theta_4$ | 258 |
| 5 | $\frac{\pi}{2}$ | 0 | $\theta_5$ | 0 |
| 6 | $-\frac{\pi}{2}$ | 0 | $\theta_6$ | 0 |

Nous avons appliqué un MGD sur chacune des articulations pour pouvoir afficher le robot avec le repère de chacune de ses articulations.



Ainsi, nous avons une fonction qui nous renvoie la matrice homogène du repère souhaité (argument d'entrée de la fonction):

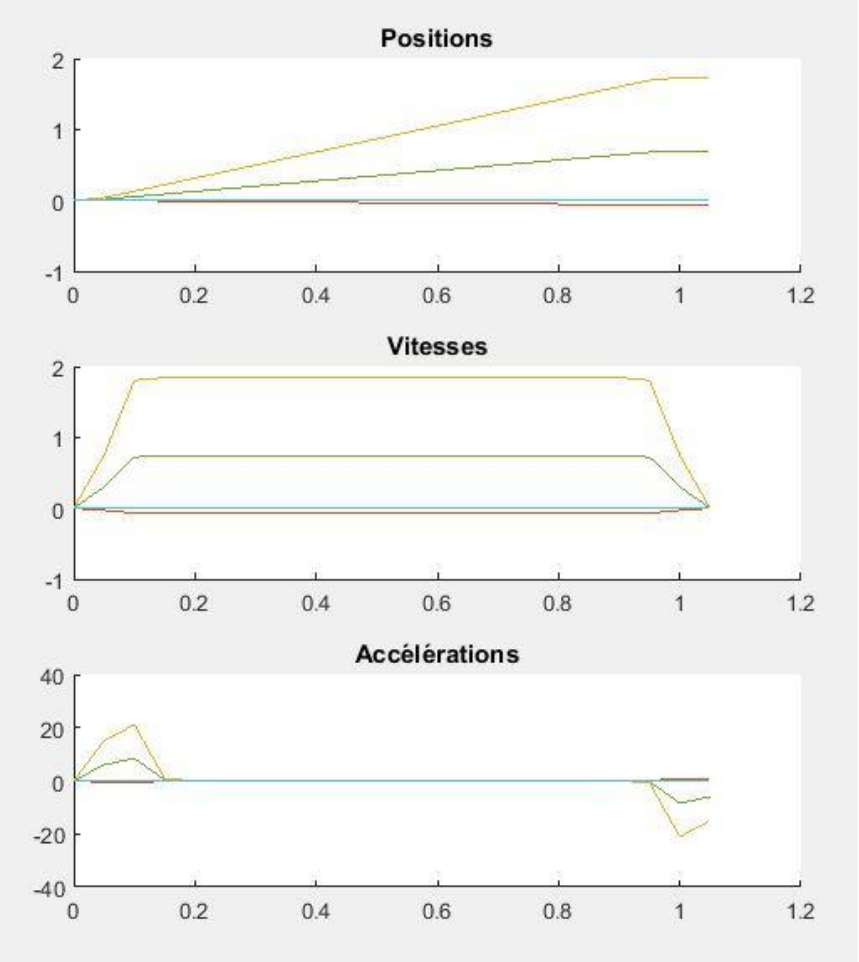
```
function T = MGD(theta, j)
% Renvoie la matrice homogène T_j
% j peut prendre les valeurs 1..8

N = 6;
if j>N+2 || j<0
    error('Invalid input : j')
end

T = eye(4);
DH=RobotisH;
theta(2)=theta(2)-1.4576;
theta(3)=theta(3)-0.8985;

for i = 1 : j*(j<=N)+N*(j>N)
    ct=cos(theta(i)); st=sin(theta(i)); d=DH.d(i); ca=cos(DH.a(i)); sa=sin(DH.a(i)); r=DH.r(i);
    T = T*[   ct   , -st     ,   0    ,    d    ;...
        ca*st , ca*ct   , -sa    ,  -r*sa   ;...
        sa*st , sa*ct   ,  ca    ,   r*ca   ;...
        0   ,   0     ,   0    ,     1    ];
end

if j==N+1 || j==N+2
    % Translation + rotation -> repère outil (comme VREP)
    T = T * [  -1.0    0.0     0.0    0.0   ;
        0.0   -1.0     0.0    0.0   ;
        0.0    0.0     1.0    123.0 ;
        0.0    0.0     0.0    1.0    ];
end

if j==N+2
    % Translation de 112mm -> pointe du crayon (comme VREP)
    T = T * [   1.0    0.0     0.0    0.0   ;
        0.0    1.0     0.0    0.0   ;
        0.0    0.0     1.0    112.0  ;
        0.0    0.0     0.0    1.0    ];
end

end
```

Pour faciliter les calculs, nous avons créé une classe "repère" qui contient toutes les données d'un repère. Nous avons aussi une classe "robot" qui contient tous les repères du robot ainsi que les lignes qui les relient entre eux.

A chaque dt, la fonction Update() permet de redessiner le robot après son déplacement.

# LA LOI TRAPÈZE

La loi trapèze est une bonne première approche de la génération de mouvement du bras robot. En effet, la courbe de vitesse de la loi trapèze est simple à analyser et permet de se familiariser avec la commande en vitesse de chaque articulation du robot.




Les articulations mettent toutes un temps $t_1$ différents avant d'arriver au "plateau de vitesse". Nous avons donc créé une fonction qui permet de calculer quelle articulation a son $t_1$ le plus grand afin de caler tous les autres $t_1$ sur le plus grand et ainsi avoir un déplacement sur chaque articulation qui commence et finit en même temps.

Cependant, nous avons trois contraintes pour la loi trapèze : une accélération et une vitesse maximale pour chaque articulation ainsi qu'une durée définie pour terminer le mouvement.

Pour pouvoir répondre au cahier des charges, nous avons divisé le problème en quatre possibilités:

1. Forme de la vitesse en triangle
    1.1. Possibilité d'arriver à la durée donnée
    1.2. Pas la possibilité d'arriver à la durée donnée
2. Forme de la vitesse en trapèze
    2.1. Possibilité d'arriver à la durée donnée
    2.2. Pas la possibilité d'arriver à la durée donnée

Si on peut arriver à la durée demandée alors on ne change rien, mais si on ne peut pas y arriver dans le temps imparti alors va à l'accélération et vitesse maximale jusqu'à arriver à la position désirée.

# LA LOI POLYNOMIALE DE DEGRÉ 5

En raison des limitations physiques des moteurs du bras, l'accélération ne peut pas passer instantanément d'une valeur à une autre. Pour surmonter ces limitations et permettre à l'accélération d'atteindre en douceur un maximum absolu, nous utilisons un polynôme de cinquième degré sous la forme de :

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

Afin de déterminer les coefficients du polynôme, nous déterminons que les conditions initiales et finales sont les suivantes :

$$\begin{aligned} q(0) &= q_i & q'(t_f) &= q_f \\ q'(0) &= 0 & q'(t_f) &= 0 \\ q''(0) &= 0 & q''(t_f) &= 0 \end{aligned}$$

En plaçant ces conditions dans l'équation polynomiale et en résolvant le système d'équations, nous obtenons les valeurs suivantes pour les coefficients polynomiaux :

$$a_0 = q_i$$
$$a_1 = 0$$
$$a_2 = 0$$
$$a_3 = \frac{10(q_f - q_i)}{t_f^3}$$
$$a_4 = -\frac{15(q_f - q_i)}{t_f^4}$$
$$a_5 = \frac{6(q_f - q_i)}{t_f^5}$$

Si la vitesse ou l'accélération dépassent les valeurs maximales fournies par le fournisseur du bras du robot, nous devons prolonger manuellement le temps nécessaire au bras pour se déplacer d'un point à un autre. C'est pour cette raison que nous cherchons à calculer à la fois la vitesse et l'accélération réelles.

Nous remarquons que la vitesse sera toujours maximale à la moitié de la durée. L'accélération maximale est une autre histoire cependant : en raison de la nature du polynôme, il se produira au moment où la dérivée de l'accélération est nulle, à τ. En termes mathématiques :

$$A' = q'''(\tau) = 6a_3 + 24a_4\tau + 60a_5\tau^2 = 0$$

Les racines de l'équation ci-dessus sont

$$\tau = \frac{-24a_4 \pm \sqrt{576a_4^2 - 1440a_3a_5}}{12a_3}$$

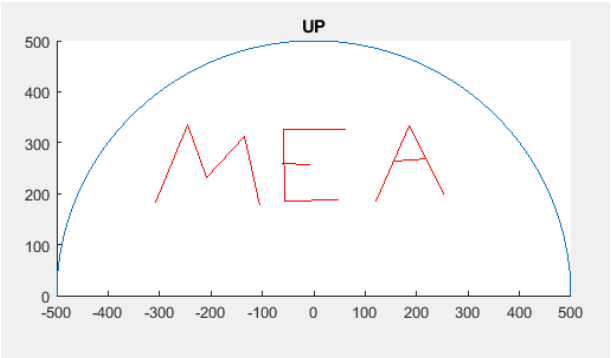
Si nous substituons maintenant les coefficients dans l'équation quadratique, nous obtenons une relation directe entre τ et $t_f$. Nous choisissons la plus petite des deux racines, car c'est le premier moment où l'accélération atteint son maximum. Nous nous retrouvons avec :

$$\tau = 0,2113 t_f$$

# COMMANDE CINÉMATIQUE

Ce type de commande permet de faire effectuer un déplacement au robot en lui donnant des positions dans le repère xyz, contrairement aux types précédents qui sont des commandes articulaires (on donne les angles souhaités pour chaque articulation).

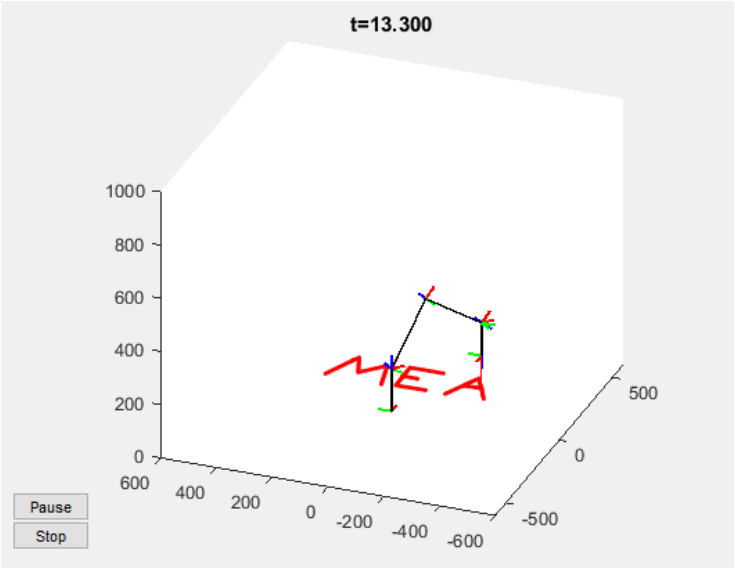
Notre code permet à l'utilisateur de dessiner sur matlab, puis le robot (à condition d'avoir défini une étape de commande cinématique) effectuera le même dessin.




Le demi-cercle bleu correspond à la surface accessible par le robot (avec le crayon à la verticale).

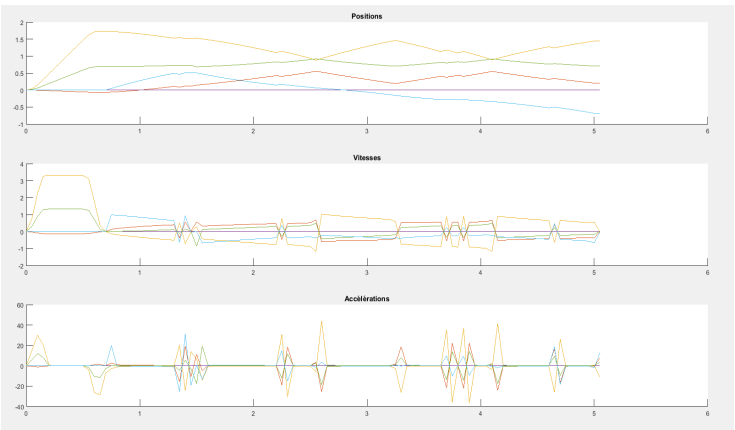
Pour dessiner il faut utiliser la souris :

- clic droit ⇔ crayon up/down (à l'état up, la pointe est levée => pas de dessin).
- clic gauche ⇔ ajout d'un point
- clic molette ⇔ quitter




Pour un bon fonctionnement de la commande cinématique, il a fallu ajuster les coefficients Kp et Ka.
Il faut que Ka soit très inférieur à Kp, car sinon la gestion de l'erreur d'orientation influe trop sur la trajectoire, et en plus le crayon oscille si Ka est trop grand.

Nous avons également ajouté une fonction qui ajoute des points à la liste de points désirées, lorsque l'écart entre les points d'origine est supérieur à 'dist'.




L'accélération n'est pas limité dans le code, on observe un léger dépassement de l'accélération max.

# CONCLUSION

Ce projet a été très intéressant car il nous a stimulé avec l'objectif de réaliser un code qui puisse contrôler le bras robotique présent au bâtiment 14. Il nous a aussi aidé à comprendre et assimiler les cours de commande en espace libre et contraint que nous avions en parallèle.

Avec un étudiant étranger au sein du groupe, nous avons donc aussi beaucoup pratiqué la communication pour lui expliquer ce qu'il nous était demandé de faire et comment nous devions le faire.