TP RMI - Trajectoires

Maxime BOUTHEROUE DESMARAIS
Thomas FOCHESATO
Florian GARDES
Abdelamnine MEHDAOUI
Luka MORAIZ
01/12/2020

Trajectoire unidimensionnelle

Les différents types de trajectoires utilisent plusieurs méthodes communes dont `getVal` qui permet d'obtenir une valeur (position, vitesse, accélération, …) en fonction du temps. Nous avons tout d'abord comparé la valeur du paramètre t avec les valeurs *start* et *end* : si ce paramètre est en dehors de ces bornes, alors la vitesse et l'accélération seront nulles (le robot n'est donc pas encore en mouvement). Pour la position, elle sera égale au premier point de contrôle pour un temps inférieur au départ et au dernier point pour un temps supérieur à *end*.

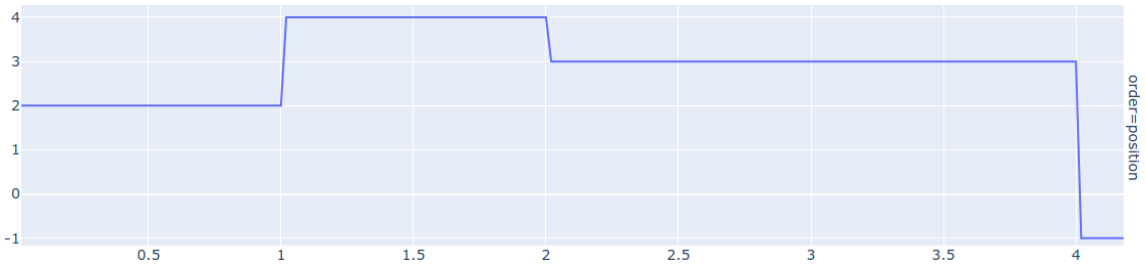
La variable *coeffs* contient une liste de polynômes, chacun permettant de relier deux points. Soient les points suivants : `A=(t0, x0) B=(t1, x1) C=(t2, x2)`. Il y aura alors deux polynômes : le premier pour les points A et B et le second pour les points B et C. Un polynôme est stocké dans une liste de taille $K+1$ où K représente le degré des polynomes, chaque indice correspond à la puissance du produit. Soit le polynôme `4x^2 + 2x + 8`, sa représentation dans une liste sera la suivante :

Indice	**Produit**
0	8
1	2
2	4

Pour les splines linéaires, la variable x sera remplacée par $t-t_i$ alors que pour les splines cubiques, elle sera uniquement remplacée par t. Pour garder une méthode générique pour toutes les splines, nous avons défini la valeur de 0 si le degré du polynôme est supérieur à 3. Une spline cubique ayant un degré de 4, $t-t_i$ reviendra à t.

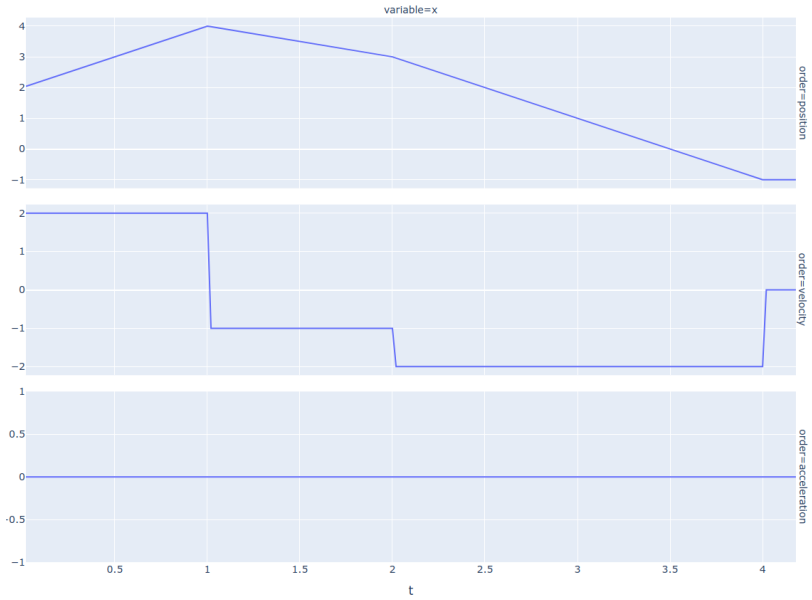
Spline constante

Pour comprendre la structure du tp, nous étions invités à implémenter la fonction pour créer une spline constante, c'est-à-dire des polynômes de degré 0 qui indiquent la position du point précédent lorsqu'une valeur de t intermédiaire lui est fournie.
Comme ce type de spline est de degré 0, la vitesse et l'accélération (respectivement dérivées première et seconde) seront nulles. La physique impliquant qu'il y ait une vitesse et une accélération pour effectuer un mouvement, ces valeurs sont sur une période extrêmement courte pour passer d'une position à une autre "instantanément". De ce fait, ce type de spline n'appartient pas à une classe de continuité, ou elle est C^n entre les points de contrôle.



Spline linéaire

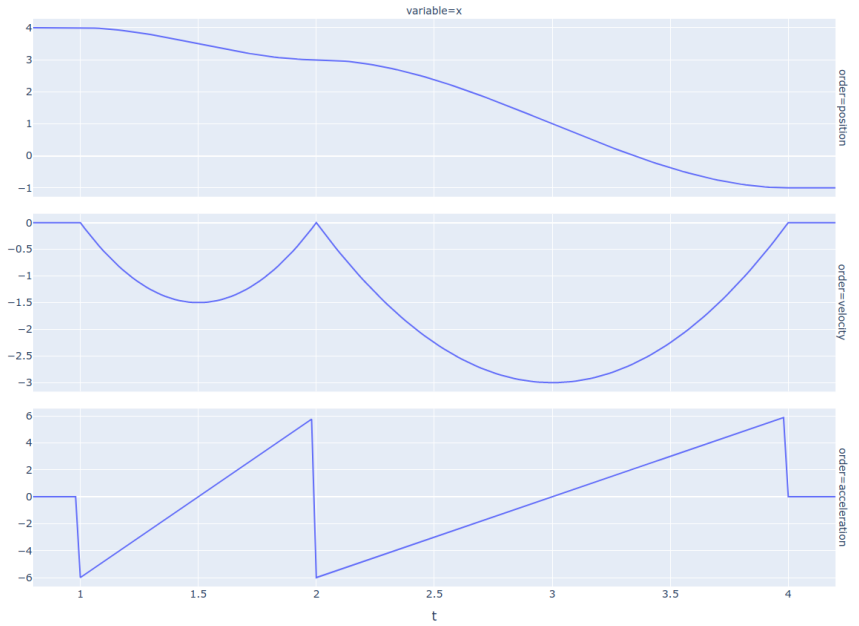
Une spline linéaire utilise des polynomes de degré 2 entre chaque points.



Elle appartient à la classe de continuité C0

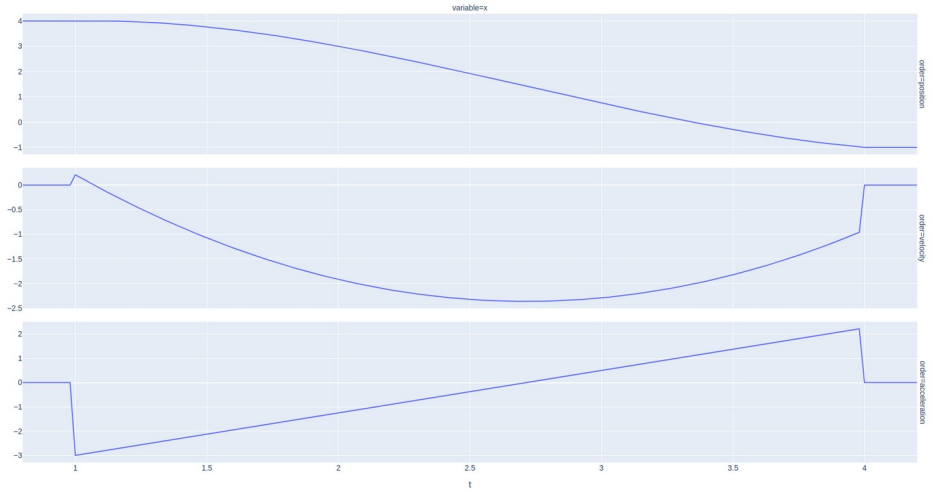
Spline cubique avec dérivées nulles aux points de passages

Ce type de spline cubique possède deux contraintes pour faire en sortie que la vitesse soit nulle à chaque point de contrôle ce qui impliquera à une décélération du robot à l'approche d'un point.



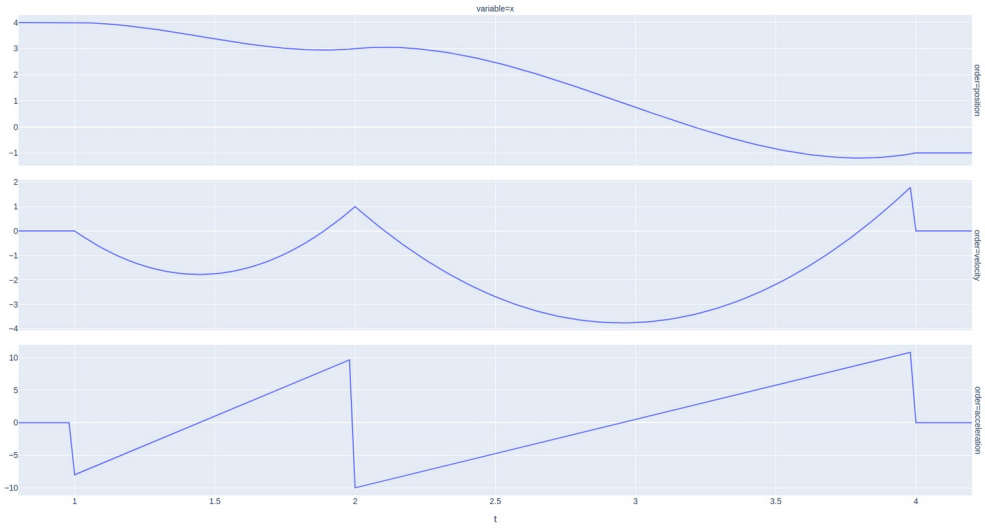
Nous pouvons constater que la vitesse est continue mais que l'accélération est infinie, la classe de continuité de cette trajectoire est donc C^0

Spline cubique basée sur un plus grand voisinage



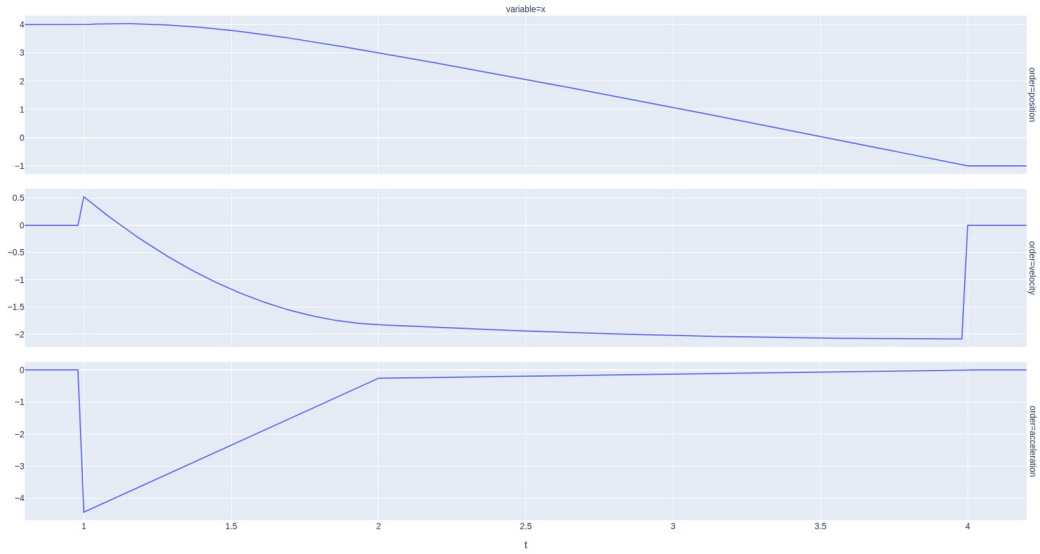
La classe de continuité de cette trajectoire est C^1

Spline cubique avec vitesse spécifiée par l'utilisateur



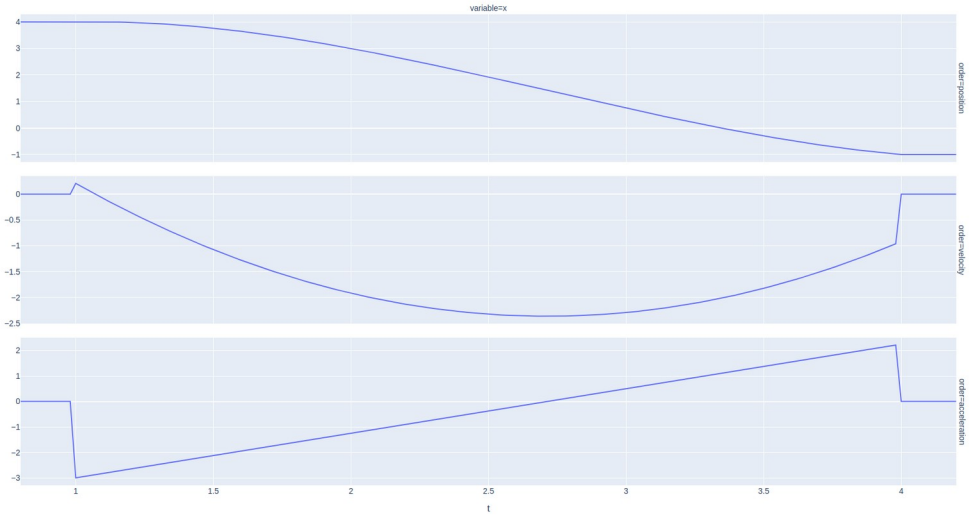
Ici on peut voir que la vitesse est continue mais que l'accélération est infinie donc cette trajectoire est C^0

Spline cubique naturelle



La vitesse et l'accélération sont continues, cette trajectoire appartient à la classe de continuité C^1

Loi trapézoidale en vitesse



Nous pouvons constater que cette trajectoire est de classe C^0 car la vitesse est continue mais que l'accélération est infini.

Trajectoires multi-dimensionelles

La seconde partie du tp nous proposait d'expérimenter la génération de trajectoires en spécifiant l'espace des cibles et des trajectoires. Dans le cas où les deux espaces sont différents, il faudra effectuer une conversion des données en utilisant le modèle MGD ou MGI suivant le sens de la conversion (opérationnel vers articulaire ou articulaire vers opérationnel). La conversion servira à transformer les cibles dans l'espace de la trajectoire.

La méthode que nous avons utilisée est constituée de trois cas :

- Si l'espace utilisé est l'espace opérationnel alors les trajectoires seront définies par quatre éléments : le temps et les coordonnées x, y et z. Il faudra créer une trajectoire pour chaque dimension vers le point cible. Lorsque l'espace articulaire est spécifié, il faudra que les points de la trajectoire possèdent autant de dimensions que le nombre d'articulations.
- Pour passer de l'espace opérationnel à l'espace articulaire, il faudra passer par le calcul du MGI. Nous utilisons pour cela ce qui a été fait au tp précédent, c'est-à-dire la méthode analytique pour la position et l'inverse de la Jacobienne pour pour la vitesse. En ce qui concerne l'accélération, nous avons voulu faire la dérivée de la Jacobienne en fonction du temps $\dot{J} = \frac{dJ}{dt} = \frac{\frac{dJ}{dq} * dq}{dt} = \frac{dJ}{dq} * \dot{q}$, on obtient alors $\ddot{q} = J^{-1} * (\ddot{x} - \dot{J} * \dot{q})$.
- Pour passer de l'espace articulaire à l'espace opérationnel, il faudra passer par le calcul du MGD. La vitesse est obtenue en calculant la jacobienne et l'accélération : $\ddot{x} = J * \ddot{q} + \dot{J} * d\dot{q}$

Méthodes de test

Nous avons réalisé différents types de tests pour réduire le nombre de bugs possible. Nous avons tout d'abord testé pour les premières splines que le comportement obtenu était celui attendu : position initiale avant le start et position finale après le end, vitesse nulle, accélération nulle ou encore test de continuité / discontinuité. Pour cette dernière méthode, nous prenons des paires de points séparés par 0.2 seconde afin de vérifier que la distance parcourue ne dépasse pas une certaine valeur. Si c'est le cas alors la courbe est discontinu. Ce test n'est pas parfait mais permet de vérifier plusieurs splines automatiquement en cas de modification de la méthode getVal qui pourrait entrainer des régressions.

Une deuxième méthode consistait à générer des knots aléatoirement ainsi que des starts différents pour vérifier que tous les points de contrôle étaient atteints pour chaque spline. Cette méthode a l'avantage de pouvoir tester des cas auxquels nous n'aurions pas pensé.