TP2 : Commande avec anticipation, commande prédictive

Dajing Gu & Alice Phe

Décembre 2020

1 Introduction

1.1 Résumé du Cours

Contrairement au modèle cinématique, le **modèle dynamique** prend en compte les forces, inerties et frottements du système et permet ainsi d'avoir une vue plus 'réaliste' et plus complexe du système. On peut ainsi prédire, d'**anticiper** le comportement du système et donc de choisir les entrées (commandes etc...) qui **minimise notre coût défini** et optimise notre utilité. Pour atteindre cette optimum de **commande prédictive**, selon la linéarité du systèmes et de ses contraintes, il existe différentes méthodes de résolution dont nous allons en voir dans le TP (NMPC, Chen et Allgower).

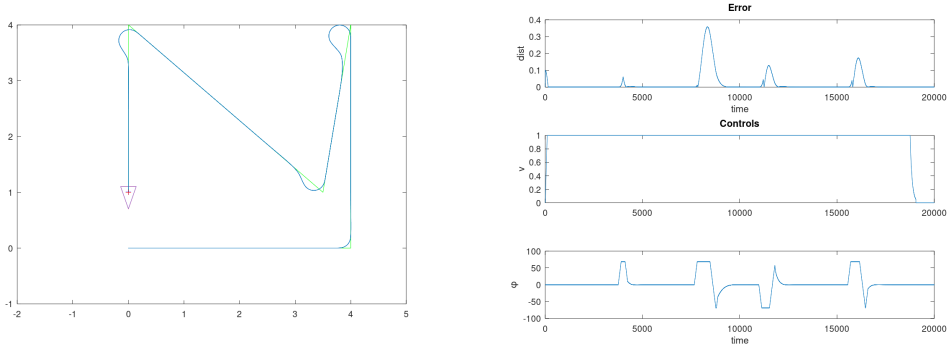
1.2 Introduction du TP

En utilisant Matlab (Octave), nous regardons l'influence de la taille de l'horizon d'anticipation dans le suivi de chemin. Nous calculons ensuite la zone de stabilité pour la commande prédictive d'une bicylette et la testons.

2 Anticipation

Dans cette section, on a comparé le résultat du contrôle du robot par PID et par MPC. K_ρ est fixé à **10** et K_α est fixé à **5** pour que la comparaison soit raisonnable.

2.1 Comparaison entre PID et MPC



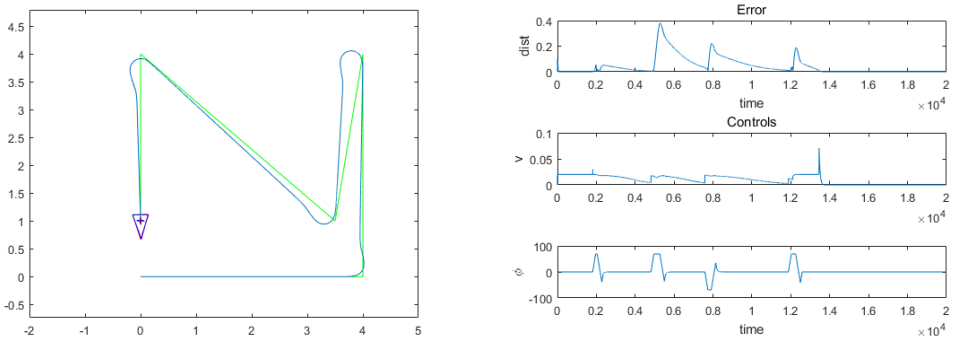
(a) Trajectoire du robot

(b) Évolution d'erreur du robot

Figure 2.1: Illustration de PID, err = 368

Selon les résultats présentés dans la figure 2.1 et la figure 2.2, on peut trouver que la trajectoire réelle du robot est plus proche du chemin prédéfini quand le robot est contrôlé par PID.

En utilisant la méthode du PID, le robot a suivi la trajectoire constituée d'une série des points sur le chemin prédéfini. Le robot se déplace toujours à coté de ce chemin, même s'il y a des oscillations. Mais pour la méthode MPC, la trajectoire n'est pas complètement sur le chemin. En effet, MPC est un contrôle optimal à long terme qui prédit le futur et s'adapte par un calcul en boucle ouverte pour arriver de façon optimale à la position finale, ie en prédisant les valeurs de la commande sur un **horizon**, et parfois sans suivre les positions intermédiaires du chemin.



(a) Trajectoire du robot

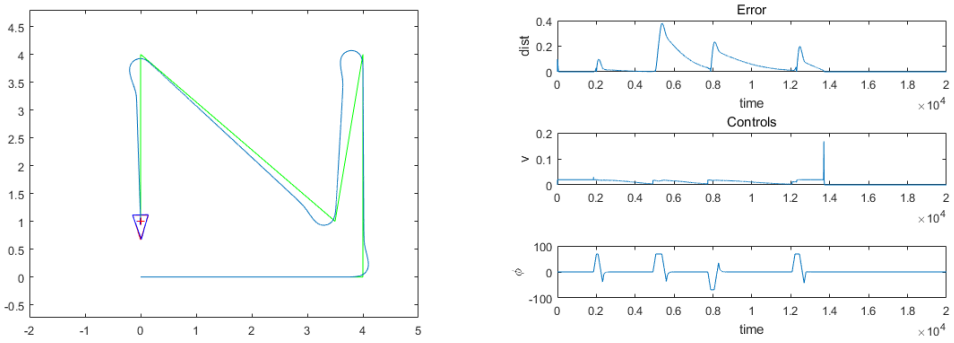
(b) Évolution d'erreur du robot

Figure 2.2: Illustration de MPC avec Window Size 5, err = 867

2.2 Essai de plusieurs horizon: 1, 5, 20, 100, 1000

En essayant des valeurs différentes de la taille de l'horizon, dont les résultats sont présentés dans les figures 2.3, 2.2, 2.5, 2.6, 2.7 on peut conclure que plus la taille de l'horizon est grande, plus l'erreur de contrôle est grande et plus la trajectoire choisie est éloignée du chemin.

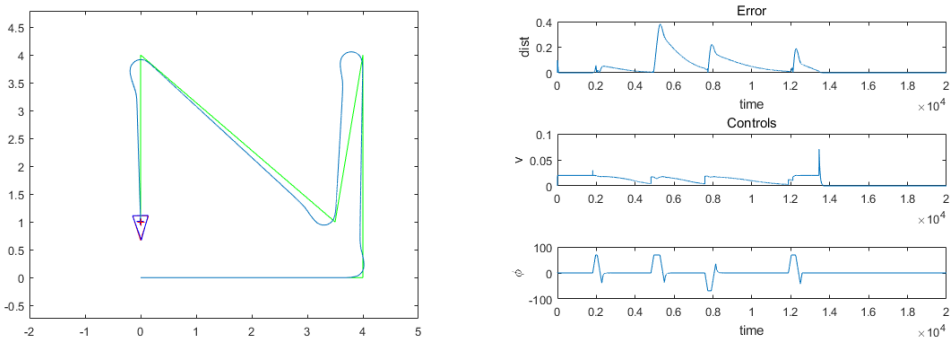
Plus grande la taille de l'horizon, plus grand le pas vers le futur avant chaque déplacement : ce qui aide le robot à se diriger vers la destination finale directement (comme dans le cas Window Size = 200 et dans le cas Window Size = 1000).



(a) Trajectoire du robot

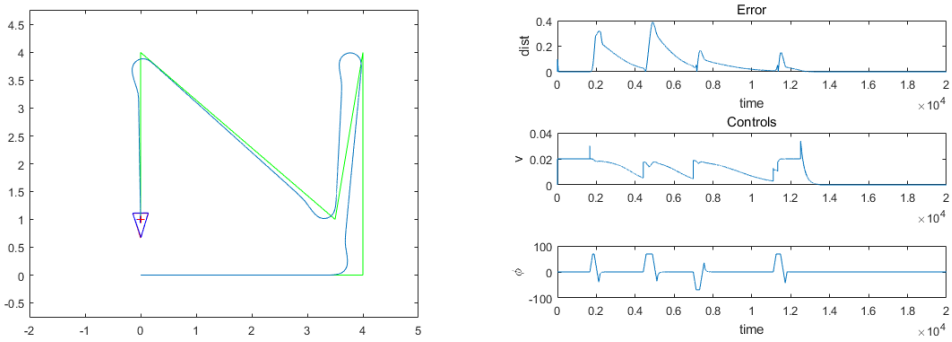
(b) Évolution d'erreur du robot

Figure 2.3: Illustration de résultat avec Window Size 1, err = 867



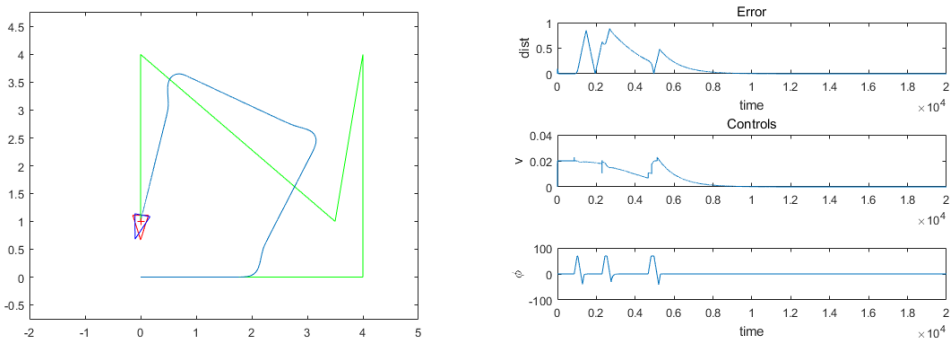
(a) Trajectoire du robot (b) Évolution d'erreur du robot

Figure 2.4: Illustration de résultat avec Window Size 5, err = 867



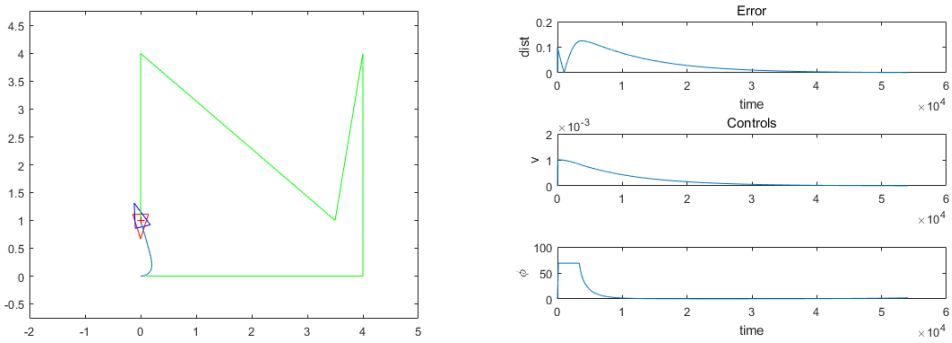
(a) Trajectoire du robot (b) Évolution d'erreur du robot

Figure 2.5: Illustration de résultat avec Window Size 20, err = 1016



(a) Trajectoire du robot (b) Évolution d'erreur du robot

Figure 2.6: Illustration de résultat avec Window Size 100, err = 2405



(a) Trajectoire du robot

(b) Évolution d'erreur du robot

Figure 2.7: Illustration de résultat avec Window Size 1000, err = 1687.9893

3 Zone de Stabilité d'une commande prédictive

Dans cette section, on étudie la zone de stabilité d'une commande prédictive avec la méthode décrite dans [1]. Après le calcul, on peut obtenir une zone de stabilité de Lyapunov cf Fig. 3.1. Ce sont toutes les positions où la condition de stabilité est vérifiée.

Le système à étudier dans ce section est (cf Section 5.Example de [1]) :

$$\begin{aligned} \dot{x}_1 &= x_2 + u(\mu + (1-\mu)x_1) \\ \dot{x}_2 &= x_1 + u(\mu - 4(1-\mu)x_2) \end{aligned} \tag{3.1}$$

En linéarisant l'équation 3.1: $\tilde{x_1} = x_1 - x_{10}$ et $\tilde{x_2} = x_2 - x_{20}$, on obtient:

$$\begin{aligned} \dot{\tilde{x}}_1 &= \tilde{x}_2 + u(1-\mu)\tilde{x}_1 + u(\mu + (1-\mu)x_{10}) + x_{20} \\ \dot{\tilde{x}}_2 &= \tilde{x}_1 - 4u(1-\mu)\tilde{x}_2 + u(\mu - 4(1-\mu)x_{20}) + x_{10} \end{aligned} \tag{3.2}$$

On en déduit que

$$A = \begin{pmatrix} u(1-\mu) & 1 \\ 1 & -4u(1-\mu) \end{pmatrix} \; et \; B = \begin{pmatrix} \mu + (1-\mu)x_{10} \\ \mu - 4(1-\mu)x_{20} \end{pmatrix}$$

```
[x, l, g] = care(A, B, Q, R); %Octave
K =- g
```

En suite, à l'aide de `care`, on peut résoudre l'équation de Riccati, $[x, l, g] =$ `care`(A, B, Q, R). En notant $K = -g$, on a $A_K = A + BK$.

```
%systeme rebouclage
Ak=A+B*K;
```

Eu puis, on calcule la borne λ (la valeur propre maximale de A_K) et la borne alpha a 95 % de λ.

```
%calcul de la borne, on retrouve bien celle de l'article
lambda=-max(eigs(Ak));
% borne a 95 %
alpha=lambda-0.05*lambda;
```

Après avoir résolu les matrices de l'équation de Lyapunov à l'aide de `lyap`, on obtient P, qui nous aide à vérifier si un point est dans la zone de stabilité du contrôleur.

```
%matrice pour equation de Lyapunov
Al=(Ak+[alpha, 0;0, alpha])';
Bl=(Q+K'*R*K);

P=lyap(Al,Bl);
```

Enfin nous trouvons β_1 la borne de la région de satisfaction des contraintes en utilisant la fonction `qp` du package control.

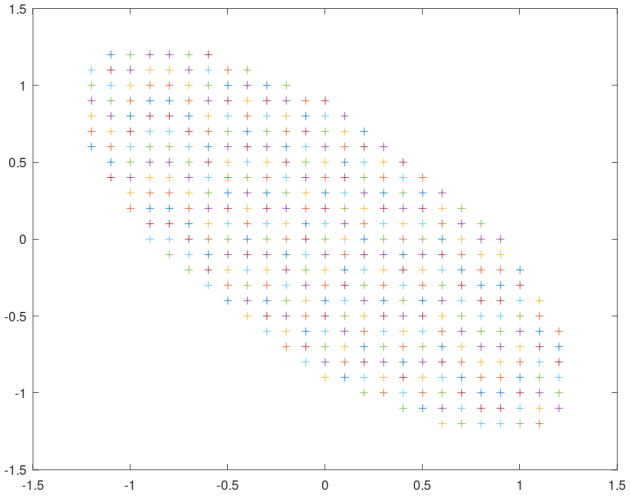


Figure 3.1: Zone de points stables

La figure 3.1 des points pour lesquels la commande est stabilisante forme une ellipse centrée en [0;0] de demi-grand axe entre [-1:1] et [-1;-1] et demi-petit axe entre [-0.5;-0.5] et [0.5;0.5], comme prédit par la définition du bassin de stabilité au voisinage de [0;0].

4 Commande Prédictive

Dans la question 2, on a déjà obtenu K = [2.118, 2.118], et on va utiliser `test_mpc.m` dans cette section pour le tester $u = Kx$ cf Fig.1(a). On peut voir que les différents points d'origine $[0.683; -0.864]$, $[-0.523; 0.244]$, $[0.808; -0.121]$, $[0.774; -0.222]$, $[0.292; -0.228]$, $[-0.08; -0.804]$ (tous dans le bassin de Lyapunov prédit) convergent vers le point de référence [0,0], avec un pas de plus en plus petit lié à la résolution quadratique du problème.

Pour obtenir la commande prédictive, on discrétise l'algorithme non linéaire à l'aide d'un schéma d'Euler et on vectorise sur un horizon de 4 pour obtenir l'équation à résoudre par une approche des moindres carrés :

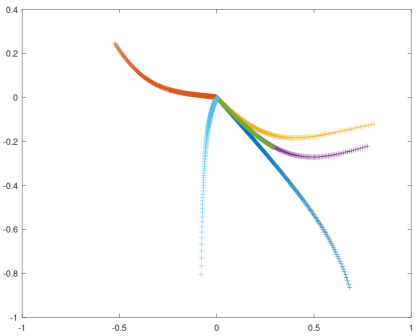
$$X(k) = \hat{A}\tilde{x}(k) + \hat{B}U$$
$$U = -\hat{B}^{\#}\hat{A}\tilde{x}(k)$$

avec

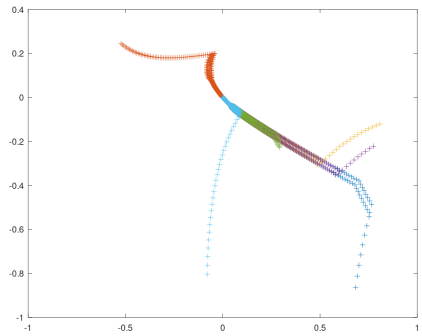
$$\hat{A} = \begin{pmatrix} A(k) \\ A^2(k) \\ A^3(k) \\ A^4(k) \end{pmatrix}, \hat{B} = \begin{pmatrix} B(k) & 0 & 0 & 0 \\ A(k)B(k) & B(k) & 0 & 0 \\ A^2(k)B(k) & A(k)B(k) & B(k) & 0 \\ A^3(k)B(k) & A^2(k)B(k) & A(k)B(k) & B(k) \end{pmatrix}$$

```
A=eye(2,2)+dt*[u*(1-mu), 1; 1, -u*4*(1-mu)];
B=dt*[mu+(1-mu)*x(1); mu-4*(1-mu)*x(2)];
Aqp=-[A;A*A;A*A*A;A*A*A*A]*x;
Bqp=[B, zeros(2, n-1); A*B, B, zeros(2, n-2); A*A*B, A*B, B, zeros(2,n-3); A*A*A*B, ...
    A*A*B, A*B, B];
U=inv(Bqp'*Bqp)*Bqp'*Aqp;
```

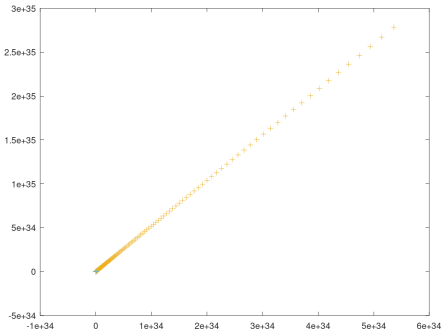
La Fig. 1(b) donne 6 simulations de trajectoires avec la commande prédictive. On peut y voir que les courbes convergent vers la position de référence mais avec plus de difficultés et la trajectoire la plus courte n'est pas choisie, et que les courbes ont tendance à vouloir revenir sur un axe menant à la position de référence.



(a) Trajectoires avec une trajectoire stabilisée



(b) Trajectoires avec commande prédictive



(c) Trajectoires avec commande prédictive - divergence des trajectoires des points hors du bassin

Figure 4.1: Résultats de `simulateMPC`

Pour atteindre la position recherchée (ici [0,0]), la méthode de [1] semble donner de meilleurs résultats que la simple commande prédictive. En effet, il s'agit de l'approche généralisée pour laquelle la fonction de coût qu'on minimise prend en compte l'état final de notre prédiction alors que l'approche simple minimise seulement une erreur à la trajectoire de référence, d'où des changements brusques de directions.

References

[1] H. Chen and F. Allgöwer, "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability," pp. 1421–1426, 1997.