Rapport Automatique

Aya Rifai
Fadwa Ellaik

Décembre 2024

Sommaire

1 Introduction

Ce rapport portera sur les travaux dirigés 2 et 3 et les travaux pratiques 2 et 3 de l'automatique. Dans l'analyse du TD2, nous montrerons le résultat d'un contrôle en boucle fermée sur la stabilité d'un système autour d'un point d'équilibre. Quant au TD3, il nous sera un support pour apprendre à stabiliser des systèmes linéaires et non linéaires par retour d'état, ce qui nous servira dans nos résultats et nos approches du TP2. Le TP02 aura donc pour but la simulation du pendule inversé contrôlé par retour d'état et de sortie. Le TP03 porte sur la simulation du robot Lego, il nous servira de support théorique pour attaquer l'approche pratique faite en TP04 pour déplacer le robot Lego.

2 Analyse du TD2:

L'objectif du TD2 était de démontrer la nécessité d'un contrôle en boucle fermée pour stabiliser un système autour d'un point d'équilibre.

2.1 Exercice1

Le 1$^{\text{er}}$ exercice traite du système :

$$\begin{cases} \ddot{\theta}(t) - \theta(t) = u(t) \\ \theta(0) = 1 \\ \dot{\theta}(0) = -2 \end{cases}$$

Nous avons alors écrit le système sous la forme :

$$\text{(IVP)} \quad \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ x(0) = x_0 \end{cases}$$

, en posant $x = (\theta, \dot{\theta})$, avec

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, x_0 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Nous nous sommes placés dans le cas d'une boucle ouverte avec un contrôle $u(t) = 3e^{-2t}$.
Nous avons alors trouvé que :

$$x(t) = e^{tA}x_0 + \frac{3}{2}e^{tA}\begin{pmatrix} e^{-t} - \frac{1}{3}e^{-3t} + \frac{2}{3} \\ -e^{-t} - \frac{1}{3}e^{-3t} + \frac{4}{3} \end{pmatrix}$$

.
Nous avons observé que le système n'était pas intrinsèquement stable, car il ne

compensait pas les perturbations persistantes. Une approche en boucle fermée était donc nécessaire.
Nous avons alors considéré le contrôle $u(t) = k_1\theta(t) + k_2\dot{\theta}(t)$. Nous avons donc écrit le système sous la forme :

$$\dot{x}(t) = Mx(t), \quad x(0) = x_0$$

où

$$M = \begin{pmatrix} 0 & 1 \\ k_1 + 1 & k_2 \end{pmatrix}$$

.
Pour que $x_e = (0, 0)$ soit asymptotiquement stable, il faut que :

$$\begin{cases} \det(M) > 0, \\ \mathrm{Tr}(M) < 0 \end{cases} \iff \begin{cases} k_1 < -1, \\ k_2 < 0. \end{cases}$$

2.2 Exercice 2

Nous avons eu recours dans cet exercice à la stabilisation avec un contrôle PID. Notre système a été modélisé par :

$$\begin{cases} \ddot{\theta}(t) - \theta(t) = u(t) + e \\ \theta(0) = 1, \quad \dot{\theta}(0) = -2 \end{cases}$$

, où e est une constante .
Après avoir introduit une variable supplémentaire au système représentant la primitive de θ en t=0 , nous avons trouvé que :

$$\lim_{t \to +\infty} x(t) = \left(-\frac{e}{k_0}, 0, 0 \right)$$

Ce résultat illustre la capacité d'un contrôle PID (Proportionnel, Intégral, Dérivé) à compenser les perturbations constantes, mais avec une erreur statique proportionnelle à e.

3 Analyse du TD3:

L'objectif du TD3 était d'apprendre à stabiliser des systèmes (linéaires ou non) par retour d'état . Le TD3 nous a alors servi pour la réalisation du TP2.

3.1 Exercice1

Le 1$^{\text{er}}$ exercice traite du système :

$$(\mathrm{S}) \begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = u(t) \end{cases}$$

Le système (S) peut alors s'exprimer par :

$$\dot{x}(t) = f(x(t), u(t)) = Ax(t) + Bu(t).$$

, en posant $\dot{x}(t) = (x_1(t), x_2(t))$, avec

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

Les points de fonctionnement de ce système sont les points (x_e, u_e) tels que

$$(x_e, u_e) = (x_1, 0, 0).$$

En utilisant le critère de contrôlabilité de Kalman, on trouve que

$$\text{rang}(B\ AB) = \text{rang} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = 2.$$

Le système est donc contrôlable. On considère le point de fonctionnement $(0, 0, 0)$ et le contrôle $u(t) = Kx(t)$. Trouvons alors $K = (k_1, k_2)$ pour que x_e soit asymptotiquement stable:
L'unique valeur propre de $A + BK$ est -1. Ainsi, on a :

$$\begin{cases} k_1 = -1 \\ k_2 = -2 \end{cases}$$

Par contre, pour $u(t) = kx_1(t)$, on trouve que $\dot{x}(t) = Cx(t)$, avec

$$C = \begin{bmatrix} 0 & 1 \\ k & 0 \end{bmatrix}.$$

On trouve que $\text{tr}(C) = 0$ et donc, dans ce cas, le système ne pourra pas être asymptotiquement stable.

3.2 Exercice2

L'exercice 2 traite de la stabilisation d'un pendule inversé que nous verrons dans le TP2.

4 TP02 : Simulation du pendule inversé contrôlé par retour d'état et de sortie

L'objectif du TP2 était de simuler le pendule inversé contrôlé par retour d'état. Le système s'écrit alors:

$$(S)\begin{cases}\dot{x}_1(t) = x_2(t),\\ \dot{x}_2(t) = \frac{g}{l}\sin(x_1(t)) - \frac{\cos(x_1(t))u(t)}{l},\\ x_1(0) = x_{0,1} = \alpha_0,\\ x_2(0) = x_{0,2} = \dot{\alpha}_0.\end{cases}$$

avec les notations mathématiques suivantes:

- $g = 9.81$;
- $l = 10$;
- $t_0 = 0$;
- $x_e = (0, 0)$;
- $u_e = 0$;
- $u(t) = u_e + K(x(t) - x_e)$;
- $K = (k_1, k_2)$;

Pour le système précédent, en posant $x(t) = (x_1(t), x_2(t))$, nous avons obtenu :

$$\begin{cases}\dot{x}(t) = (\dot{x}_1(t), \dot{x}_2(t)) = \left(x_2(t), \frac{g}{l}\sin(x_1(t)) - \frac{\cos(x_1(t))u(t)}{l}\right) = f(x(t), u(t))\\ x_0 = (x_1(0), x_2(0)) = (\alpha_0, \dot{\alpha}_0)\end{cases}$$

avec

$$f : \mathbb{R}^3 \to \mathbb{R}^2$$
$$(x, u) \mapsto \begin{pmatrix} x_2 \\ \frac{g}{l}\sin(x_1) - \frac{\cos(x_1)u}{l}\end{pmatrix}$$

Nous avons choisi un contrôle $u(t)$ par retour d'état. C'est une méthode de contrôle des systèmes dynamiques où l'entrée $u(t)$ est calculée en fonction de l'état complet du système. Le contrôle par retour d'état sert à placer le système dans des positions qui garantissent une convergence rapide vers la position ou la trajectoire désirée en utilisant uniquement les mesures d'état disponibles. Elle est utilisée pour les systèmes linéaires et étendue à des systèmes non linéaires. C'est le cas du système (S) dont nous traitons où f est une fonction non linéaire.

Ainsi ,pour le contrôle $u(t) = u_e + K(x(t) - x_e)$, le système s'écrit :

$$\begin{cases}\dot{x}(t) = g(x(t))\\ x_0 = (\alpha_0, \dot{\alpha}_0)\end{cases}$$

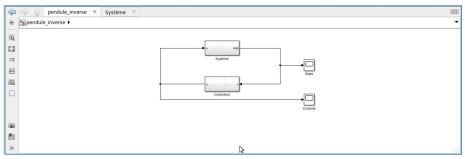

Figure 1: Schéma SIMULINK.

avec

$$g : \mathbb{R}^2 \to \mathbb{R}^2$$
$$x \mapsto f(x, u)$$

De ce,

$$g'(x_e) = \frac{\partial f}{\partial x}(x_e, u_e) + \frac{\partial f}{\partial u}(x_e, u_e)K =: A + BK.$$

Donc, pour contrôler asymptotiquement le système (S) pour $(\alpha_0, \dot{\alpha}_0)$ suffisamment proche de x_e, il faut trouver K telle que :

$$\mathrm{sp}(g'(x_e)) \subset \mathbb{R}_-^*$$

On a :

$$\mathrm{Jac}(g(x_e)) = \begin{pmatrix} 0 & 1 \\ \frac{(g-k_1)}{l} & -\frac{k_2}{l} \end{pmatrix}$$

Comme la matrice est dans $M_2(\mathbb{R})$, le système est asymptotiquement contrôlable si et seulement si :

$$\begin{cases} \mathrm{tr}(\mathrm{Jac}(g(x_e))) < 0 \\ \det(\mathrm{Jac}(g(x_e))) > 0 \end{cases} \iff \begin{cases} -\frac{k_2}{l} < 0 \\ \frac{g-k_1}{l} < 0 \end{cases} \iff \begin{cases} k_1 > g \\ k_2 > 0 \end{cases}$$

Donc, pour

$$\begin{cases} k_1 > g \\ k_2 > 0 \end{cases}$$

le système est asymptotiquement contrôlable.

Nous avons réalisé le schéma SIMULINK correspondant au système :

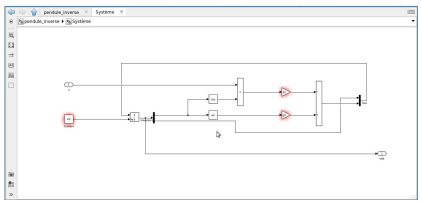

Figure 2: Schéma SIMULINK du système.

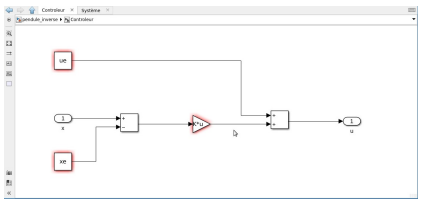

Figure 3: Schéma SIMULINK du contrôleur.

Pour la Table 1 représentée ci-dessous (1),

Cas	x_0	t_f	K	**Intégrateur**
Cas 1.1	$\left(\frac{\pi}{20}, 0\right)$	10	(30, 10)	ode45
Cas 1.2	$\left(\frac{\pi}{20}, 0\right)$	100	(10, 1)	ode45
Cas 1.3	$\left(\frac{\pi}{20}, 0\right)$	100	(10, 1)	Euler, ode1
Cas 1.4	$\left(\frac{\pi}{20}, 0\right)$	1000	(10, 1)	Euler, ode1
Cas 1.5	$\left(\frac{\pi}{10}, 0\right)$	100	(10, 1)	ode45
Cas 1.6	$\left(\frac{\pi}{10}, 0\right)$	100	(30, 10)	ode45

Table 1: Données pour le contrôle par retour d'état.

nous avons visualisé les résultats des différents cas:

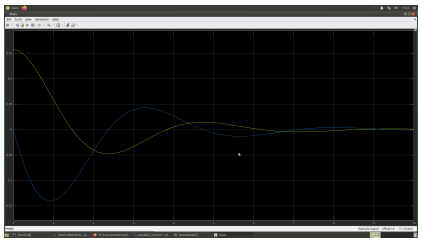

Figure 4: Résultat des état et contrôle du système pour le cas 1.1.

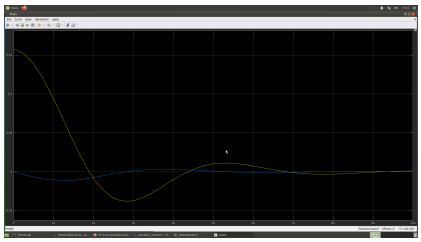

Figure 5: Résultat des état et contrôle du système pour le cas 1.2.

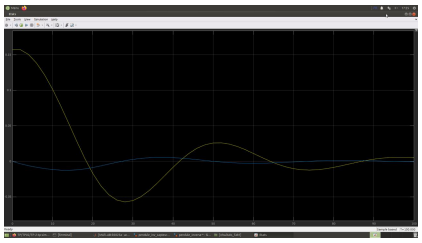

Figure 6: Résultat des état et contrôle du système pour le cas 1.3.

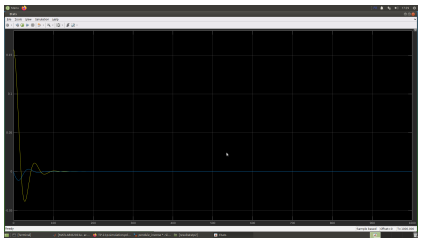

Figure 7: Résultat des état et contrôle du système pour le cas 1.4.

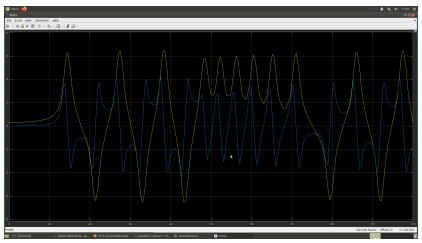

Figure 8: Résultat des état et contrôle du système pour le cas 1.5.

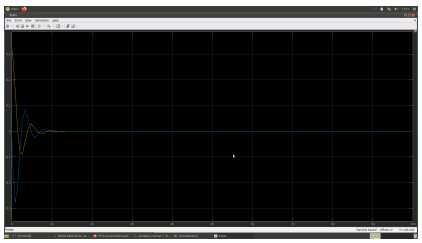

Figure 9: Résultat des état et contrôle du système pour le cas 1.6.

Comparaison des cas 1.1 et 1.2: Nous remarquons que seules les valeurs de K ((30,10) pour le cas 1.1 et (10,1) pour le cas 1.2) et de t_f(10 pour le cas 1.1 et 100 pour le cas 1.2) changent. La diminution de K entraîne une augmentation du temps de convergence. Pour la $1^{\text{ère}}$ figure,la stabilisation de l'état se fait au bout de 10 s ,alors que pour la $2^{\text{ème}}$ la stabilisation se fait au bout de 100 s.Ce qui montre l'influence de K sur le temps de convergence.
Comparaison des cas 1.2 et 1.3: Nous remarquons qu'il y a un changement d'intégrateur entre les deux figures. Pour la $2^{\text{ème}}$ figure un intégrateur ode45 est utilisé,la courbe est plus lisse et converge mieux . Pour la $3^{\text{ème}}$ figure un intégrateur Euler ode1 est utilisé,les oscillations de la courbe sont moins régulières. L'intégrateur ode45 donne des résultats plus précis et stables que l'intégrateur Euler ode1,car la $1^{\text{ère}}$ méthode adapte dynamiquement le pas de temps pour limiter les erreurs.
Comparaison des cas 1.3 et 1.4: Dans ce cas,seul t_f change(100 pour la $3^{\text{ème}}$ figure et 1000 pour la $4^{\text{ème}}$ figure). Nous remarquons alors qu'avec l'intégrateur Euler ode1, l'imprécision augmente avec l'augmentation du temps final,puisque pour la $4^{\text{ème}}$ on remarque plus d'irrégularités dues à l'amplification des erreurs numériques par l'intégrateur Euler ode1 au fil du temps.
Comparaison des cas 1.4 et 1.5: En augmentant la valeur de la condition

initiale , nous remarquons que le système ne se stabilise pas pour la $4^{\text{ème}}$ figure puisque le système s'est éloigné de son point de fonctionnement pour K fixé à (10,1).
Comparaison des cas 1.5 et 1.6: Par contre,en augmentant la valeur de K à (30,10), le système se stabilise comme nous l'avions remarqué pour les $1^{\text{ère}}$et $2^{\text{ème}}$ figures (plus K augmente plus la stabilité est garantie).

On suppose désormais qu'on a accès uniquement à $\dot{\alpha}$. On introduit alors un capteur et un prédicteur pour reconstruire α.
Nous avons alors réalisé le schèma SIMULINK representé ci-dessous :

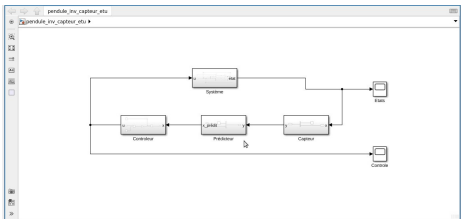

Figure 10: Schéma SIMULINK.

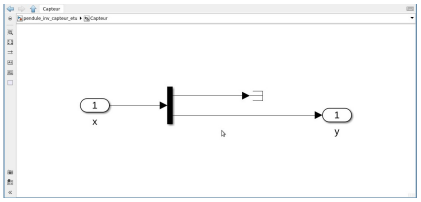

Figure 11: Schéma SIMULINK du capteur.

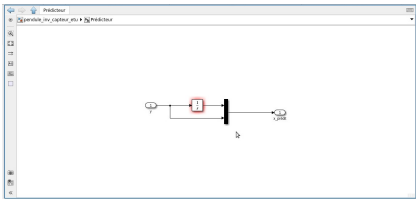

Figure 12: Schéma SIMULINK du prédicteur.

Nous avons ensuite visualisé les résultats de la Table 2 (2):

Cas	x_0	t_f	**K**	**Pas**	**Intégrateur**
Cas 1	$(\frac{\pi}{20}, 0)$	100	(10, 1)	Par défaut	ode45
Cas 2	$(\frac{\pi}{20}, 0)$	100	(10, 1)	0.001	Euler, ode1
Cas 3	$(\frac{\pi}{20}, 0)$	100	(10, 1)	5	Euler, ode1

Table 2: Données pour le contrôle par retour de sortie.

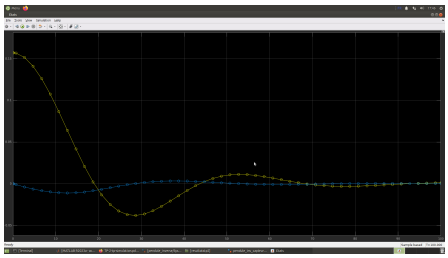

Figure 13: Résultat des état et contrôle du système pour le cas 2.1.

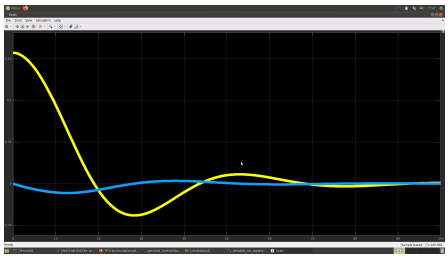

Figure 14: Résultat des état et contrôle du système pour le cas 2.2

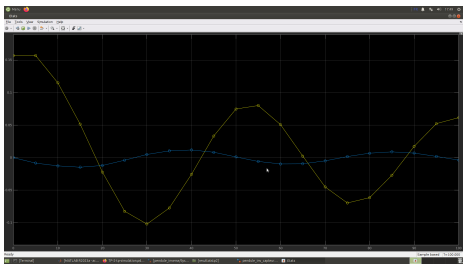

Figure 15: Résultat des état et contrôle du système pour le cas 2.3.

Comparaison des cas 2.1 et 2.2: Nous remarquons que les courbes sont assez similaires (bien qu'une soit représentée en gras et l'autre non),même avec deux intégrateurs différents, l'intégrateur ode45 étant plus précis que l'intégrateur Euler ode1. Nous pensons que cela est dû au pas pris pour l'intégrateur Euler ode1 qui est assez petit (0.001 s) . Ainsi, les irrégularités de cet intégrateur n'apparaissent pas clairement sur la courbe.
Comparaison des cas 2.2 et 2.3: Les irrégularités deviennt plus perceptibles en prenant un grand pas (5 s). Nous remarquons alors que, pour la $9^{\text{ème}}$, la convergence ne s'établit pas au bout de 100 s et les courbes montrent un comportement bruité, avec des segments très anguleux . Cela est principalement dû à la nature instable de la méthode d'Euler pour les grands pas.

5 TP03: Modèle continu et discret du robot Lego avec contrôleur et prédicteur/estimateur

L'objectif du TP3 était d'étudier le modèle du robot Lego pendule inversé continu et discret,de concevoir un contrôleur par retour d'état,d'introduire et modéliser un capteur et un prédicteur et de construire un modèle hybride. Ici, l'équation différentielle s'écrit :

$$\dot{x}(t) = f(x(t), u(t))$$

, avec $f : (x, u) \mapsto f(x, u)$.

5.1 Modèle continu:

Nous avons utilisé MATLAB pour coder la fonction $f : (x, u) \mapsto f(x, u)$ dans le fichier "fonction.m". Cette fonction nous a permis de construire un modèle continu du robot Lego à pendule inversé, comprenant un système et un contrôleur par retour d'état. Nous trouvons ci-dessous le schéma SIMULINK :

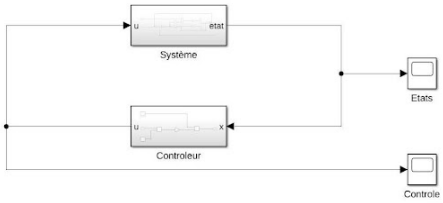

Figure 16: Schéma SIMULINK du modèle continu

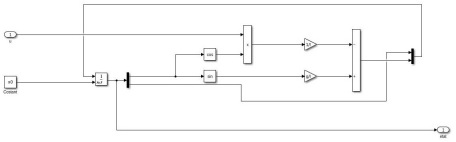

Figure 17: Schéma du système.

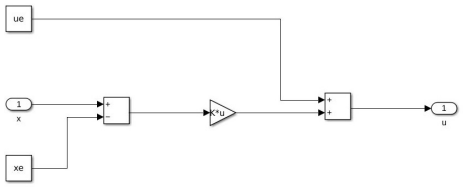

Figure 18: Schéma du contrôleur

Pour réaliser la synthèse du contrôleur par retour d'état et déterminer son coefficient K, nous avons linéarisé le système et obtenu les matrices A et B(telles que

$$\dot{x}(t) = Ax(t) + Bu(t)$$

) à l'aide du fichier MATLAB "matrices.m". En utilisant les valeurs propres de A (obtenues à l'aide de la fonction MATLAB eig, qui retourne la matrice diagonale des valeurs propres et la matrice des vecteurs propres associés), et le vecteur des valeurs propres

$$V = \begin{bmatrix} -136.5905 & -2.6555 & -3.5026 & -5.9946 \end{bmatrix}$$

, nous avons employé la fonction place(A, B, V) pour calculer $-K$ avec K vérifiant

$$u(t) = Kx(t)$$

.
On simule ce système afin d'obtenir les courbes d'état et de contrôle pour les différents paramètres donnés dans la table ci-dessous(3):

Cas	x_0	t_f	K	**Intégrateur**
Cas 1.1	$(0, 0, 0, 0)$	2	K	ode45
Cas 1.2	$(0, \pi/5, 0, 0)$	5	K	ode45
Cas 1.3	$(0, \pi/10, 0, 0)$	5	K	ode45

Table 3: Données pour le modèle du robot Lego pendule inversé.

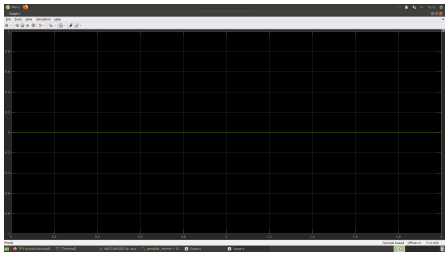

Figure 19: Résultat de l'état du système pour le cas 1.1.

Figure 20: Résultat du contrôle du système pour le cas 1.1.

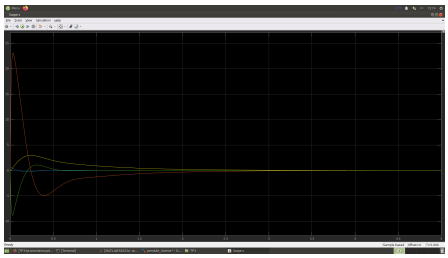

Figure 21: Résultat de l'état du système pour le cas 1.2.

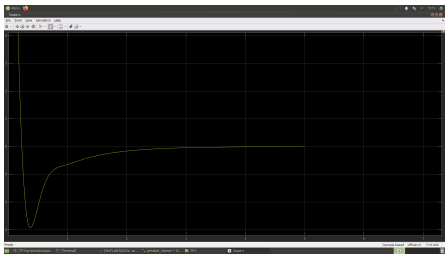

Figure 22: Résultat du contrôle du système pour le cas 1.2.

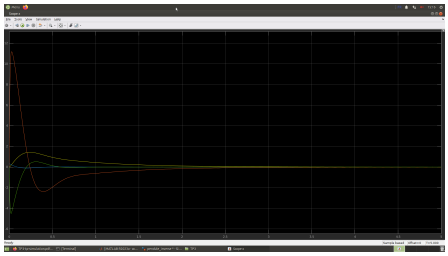

Figure 23: Résultat de l'état du système pour le cas 1.3.

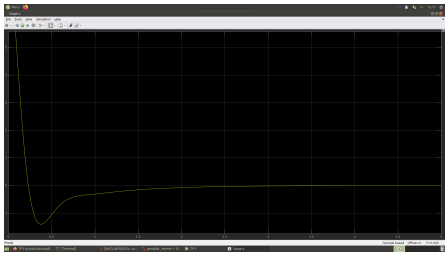

Figure 24: Résultat du contrôle du système pour le cas 1.3.

Pour le **Cas 1.1**, le système reste en équilibre avec $u(t)$ et $x(t)$ égaux à zéro, indiquant qu'aucune intervention n'est nécessaire pour maintenir la stabilité. Pour les **Cas 1.3** et **Cas 1.3**, une perturbation initiale $x_0 \neq (0, 0, 0, 0)$ entraîne une réaction active du contrôle $u(t)$ pour stabiliser le système. Les courbes montrent une stabilisation progressive des états $x(t)$ vers un point d'équilibre, reflétant un comportement stabilisant du système.

5.2 Introduction des capteur et prédicteur:

-Modélisation des capteurs (gyroscope et capteur d'angle):
Pour modéliser l'état du système, nous avons utilisé un gyroscope qui nous a permis de mesurer la vitesse de changement d'angle du corps du robot notée $\dot{\psi}(t)$ avec un capteur qui mesure l'angle $\theta(t)$.
-Développement d'un prédicteur pour recalculer les états manquants:
Les capteurs employés ne suffisent pas à modéliser l'état complet du système, puisqu'ils ne fournissent pas l'intégralité des composantes de l'état. Par

conséquent, nous avons eu recours à un sous-système prédicteur. Ce dernier estime les informations manquantes à partir des informations disponibles.

Nous avons modélisé le capteur et le prédicteur dans Simulink, puis nous avons simulé le modèle de contrôle pour chaque cas de la table suivante(4):

Cas	x_0	t_f	K	**Intégrateur**
Cas 1.1	$(0, 0, 0, 0)$	2	K	ode45
Cas 1.2	$(0, \pi/5, 0, 0)$	5	K	ode45
Cas 1.3	$(0, \pi/10, 0, 0)$	5	K	ode45

Table 4: Données pour le modèle du robot Lego pendule inversé.

Problème identifié :
Le sous-système actuel agit uniquement en fonction de l'état instantané du système sans tenir compte des états précédents. Cela entraîne des actions parfois trop brusques, réduisant ainsi la stabilité du système.
Une approche discrète a été adoptée pour moduler les actions du système, pour permettre une stabilisation progressive, mieux adaptée à la dynamique du robot.

5.3 Construction du modèle hybride:

Nous avons implémenté le contrôleur et le prédicteur en logiciel dans le robot Lego, passant ainsi d'un modèle continu à un modèle discret. Cependant, le système permettant de représenter la physique du robot reste un modèle continu. Pour ce faire, nous avons intégré un bloc Zero-Order Hold (disponible dans l'onglet Discrete de la bibliothèque Simulink) dans le capteur. Cet état reconstruit en sortie sera donc discret.
Nous montrons ci-dessous le schéma SIMULINK correspondant :

Figure 25: Schéma SIMULINK du modèle hybride

Nous avons également adapté le prédicteur pour pouvoir employer des opérateurs discrets implémentés à partir de blocs élémentaires ou de code Matlab.

Enfin, nous avons simulé le modèle de contrôle ainsi réalisé. Pour :

$$\begin{cases} g = 9.81 \\ l = 10 \\ x_e = (0;0;0;0) \\ u_e = 0 \\ K = (10,10,10,10) \\ t_f = 100 \text{ s} \\ x_0 = (0, \frac{\pi}{10}, 0, 0) \end{cases}$$

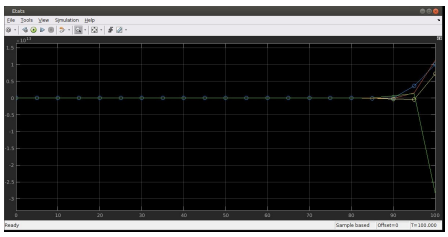

Figure 26: Résultat de l'état du système pour le modèle hybride.

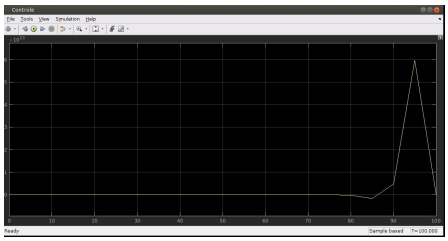

Figure 27: Résultat du contrôle du système pour le modèle hybride.

Les courbes montrent une stabilité initiale suivie d'une divergence brutale vers 90 s, ce qui indique l'instabilité du système. Les états deviennent donc incontrôlables.
En revanche, le contrôle $u(t)$ présente un pic important, révélant une tentative de rattrapage inefficace. Cela suggère un problème de discrétisation ou de gains K mal adaptés.

6 Conclusion

L'ensemble des travaux dirigés et pratiques nous a permis d'approfondir nos connaissances sur l'automatisation d'un système donné. Si le 2ème TD nous a sensibilisés sur l'importance d'un contrôle en boucle fermée et de l'intégration d'un PID pour la stabilisation d'un système donné, alors le 3ème nous a appris comment stabiliser un système linéaire ou non à l'aide d'un contrôle par retour d'état, et nous a introduits à la fonction de Liapounov permettant de montrer la stabilité des systèmes linéaires autour d'un point de fonctionnement hyperbolique.
Le TP02 nous a aidés à simuler le système régissant le mouvement d'un pendule inversé afin de pouvoir le contrôler par retour d'état. Nous avons donc constaté, à partir des différentes courbes d'état et de contrôle, que l'augmentation du vecteur K permet au système de se stabiliser plus rapidement. Nous avons aussi établi la distinction entre un intégrateur `ode45` et un intégrateur `Euler ode1`, le premier étant plus précis que le deuxième qui accumule les erreurs de calcul jusqu'à éventuellement entraîner le système vers des comportements irréguliers, voire divergents pour un grand pas.
Le TP03 nous a permis de simuler le modèle du robot Lego pendule inversé que nous avons utilisé dans la dernière séance du TP. Nous avons d'abord commencé par un modèle continu. Ainsi, nous avons modélisé le robot Lego pendule inversé dans SIMULINK à l'aide de la fonction donnée , d'abord, en utilisant uniquement un contrôleur par retour d'état , puis , en introduisant un capteur et un prédicteur , puisque l'état du système était observé par des capteurs qui ne restituaient pas l'intégralité de ses composantes.
L'introduction d'un sous-système prédicteur s'est donc avérée nécessaire pour recalculer les informations manquantes. Nous avons ensuite opté pour un modèle hybride dont l'intérêt était de combiner les avantages des modèles continus qui permettent de représenter avec précision les dynamiques physiques du système, et des modèles discrets qui sont plus adaptés pour une implantation en logiciel . Le modèle hybride nous a donc facilité la simulation dans SIMULINK, en combinant les blocs discrets et continus.