Année en cours : 2023/2024

Professeur encadrant : Rivière Peter

Bureau d'étude
Automatique

Auteurs

Ikram Berroug

Myriam Robbana

Filière / Groupe

Science du Numérique

Groupe 'F'

Table des matières

Introduction ……...3

Simulation du pendule inversé contrôlé par retour d'état…….4

Simulation du pendule inversé contrôlé……...4

Problème ……...4

Contrôle par retour d'état …….....4

Capteurs …….……...8

Simulation du robot Lego ……10

Modèle continu ……...10

Introduction des capteurs et actionneurs …….12

Construction du modèle hybride ……13

Conclusion……..15

Introduction

Le contrôle par retour d'état consiste à ajuster l'état du système pour le conduire vers une configuration souhaitée. Ainsi, cette méthode, particulièrement utilisée pour stabiliser des systèmes dynamiques, sert à s'opposer aux mouvements et favorise l'atteinte de l'équilibre, état souvent désiré. Dans ce BE, on s'intéressera à ce type de contrôle en l'appliquant d'abord au pendule inversé où l'état est entièrement défini par deux variables, puis au robot Lego qui requiert quatre variables pour caractériser son état. Ces notions ont été mises en pratique lors du dernier TP consistant à la réalisation d'un programme en langage C permettant de stabiliser un vrai robot Lego.

I. Simulation du pendule inversé contrôlé par retour d'état

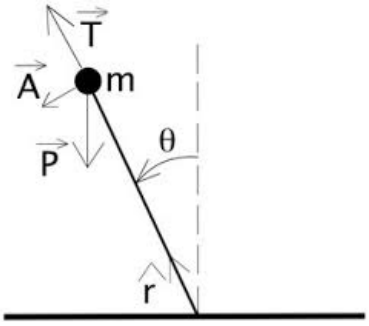
1. Simulation du pendule inversé contrôlé

1.1. Problème

Partie théorique

Objectif : Simuler le pendule inversé contrôlé par retour d'état.
On considère le système (S) suivant :

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ x_2(t) = \frac{g}{l}\sin(x2(t) - \frac{\cos(x1(t))u(t)}{l} \\ x_1(0) = x_{0,1} = \alpha_0 \\ x_2(0) = x_{0,2} = \dot{\alpha}_0 \end{cases}$$



Notations mathématiques

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

En partant du système (S), on définit la fonction f par

$$f : \mathbb{R}^2 \times \mathbb{R} \rightarrow \mathbb{R}^2$$
$$(z, v) \rightarrow (z_2, -\frac{g}{l}\sin(z1) + \frac{v}{ml2})$$

1.2. Contrôle par retour d'état

Afin de contrôler asymptotiquement le système, si $(\alpha_0, \dot{\alpha}_0)$ est suffisamment proche de x_e, il suffit que $k_1 > g$ et $k_2 > 0$.

Pourquoi ?

Ces conditions sont cruciales pour garantir que le contrôleur ait une influence suffisante sur le système pour contrer les effets de la gravité et amortir les oscillations, conduisant ainsi à un comportement stable du système.

Partie expérimentale

A l'aide du logiciel MATLAB, on réalise le schéma Simulink du modèle. En effet, la mise en œuvre dans Simulink permettra d'observer le système en action et de vérifier le bon fonctionnement du contrôleur.

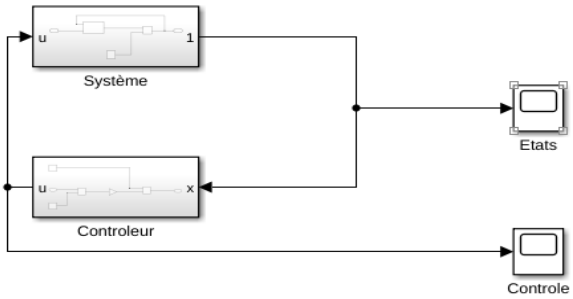
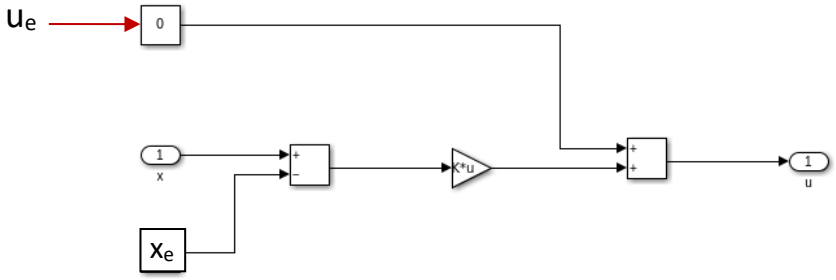


Figure 0 : Schéma n°1 Simulink

Contrôle par retour d'état

Le système évolue en prenant en compte des conditions initiales données. Le contrôleur évalue l'état du système en analysant la sortie du système, et va essayer d'agir en conséquence afin de faire en sorte que le système réponde comme l'utilisateur le souhaite, et ce, malgré les éventuelles perturbations. Dans son bloc, on représente l'équation mathématique $u(t) = u_e + K(x(t) - x_e)$.
Ainsi, le contrôleur ajuste sa sortie u, qui est l'entrée du système, pour répondre aux exigences de performance. Le système, réagit à cette réponse, et évolue en conséquence. La boucle se répète jusqu'à atteinte du résultat souhaité.

Bloc Contrôleur



On voit bien dans ce schéma que nous avons décidé de choisir $x_e = [0\ 0]$' qui signifie que la position et vitesse du pendule sont nulles, et $u_e = 0$ qui nous permet de considérer que le système est en équilibre sans action de commande externe.

De plus comme $f((x_e, u_e) = 0$, alors x_e est un point d'équilibre pour le contrôle u_e. De plus, ces différents choix, permettent de simplifier l'équation de contrôle qui se résume désormais à $u(t) = Kx(t)$

Simulation

Cas	x_0	t_f	K	Intégrateur
Cas 1.1	(π/20, 0)	10	(30, 10)	Ode45
Cas 1.2	(π/20, 0)	100	(10, 1)	Ode45
Cas 1.3	(π/20, 0)	100	(10, 1)	Euler, ode1
Cas 1.4	(π/20, 0)	1000	(10, 1)	Euler, ode1
Cas 1.5	(π/10, 0)	100	(10, 1)	Ode45
Cas 1.6	(π/10, 0)	100	(30, 10)	Ode45

Table 1 – Données pour le contrôle par retour d'état

⇨ Pour chacun des six cas, nous allons déterminer si les données initiales sont bonnes ou pas.

Résultats et Observations

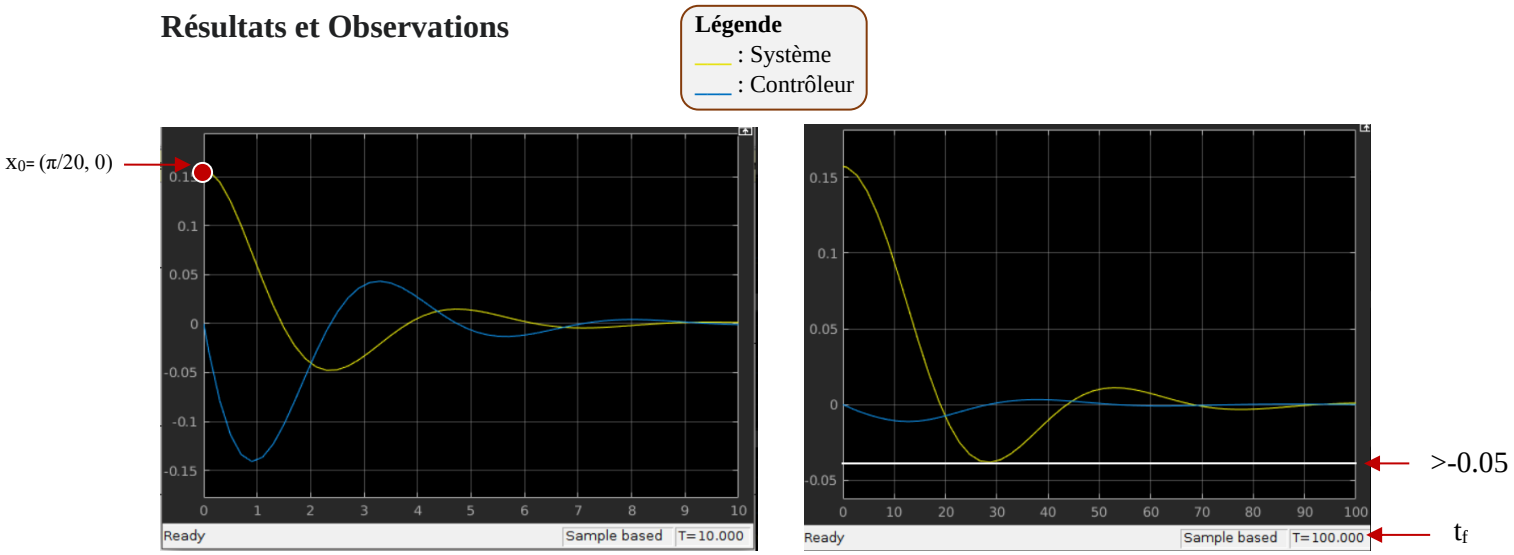


Figure 1 : Cas 1.1

Figure 2 : Cas 1.2

Observations Cas 1.1

On observe des oscillations amorties, passant de π/20 à 0 pour le système, ce qui permet de confirmer que le système converge bien vers un état stable au fil du temps.
Ceci est assez cohérent étant donné que l'on a bien k1>g et k2>0, ce qui permet d'assurer que le contrôle est asymptotiquement stable, et qu'on à bien une stabilisation vers la valeur que l'on souhaite '0'.
On remarque d'ailleurs que l'allure du contrôleur s'oppose toujours à celle du système, pour forcer la convergence vers 0. D'ailleurs, on note que l'opposition se fait avec un léger décalage qui peut être expliqué par le fait qu'il y a un temps mis pour que le contrôleur comprenne que le système a changé d'angle, et agit par la suite pour contrer ce changement d'angle. De plus, le temps tf = 10s est bien choisi étant donné qu'il permet de bien observer l'ensemble de l'évolution du système, jusqu'à sa convergence vers l'état stable.

Conclusion Cas 1.1

L'allure de la figure obtenue correspond bien à nos attentes d'un système convergeant vers une position d'équilibre. Les paramètres choisis permettent une bonne visualisation de ce phénomène.

Observations Cas 1.2

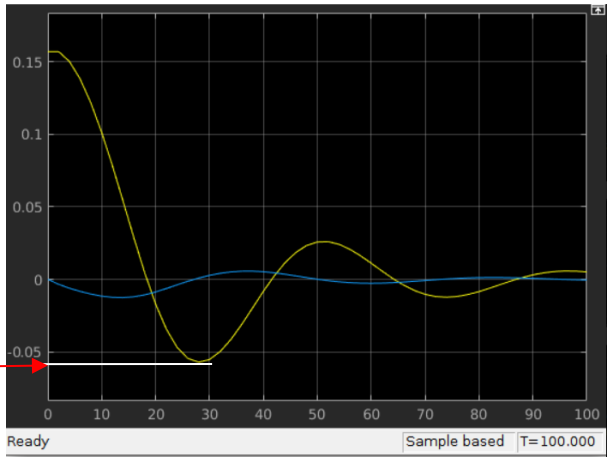
Une fois de plus, on observe bien des oscillations amorties qui sont dues au fait que k1>g et k2>0.
Nous sommes exactement avec les mêmes conditions initiales que pour le cas 1.1 mise à part pour les valeurs de K qui sont choisies plus petites.
On remarque ainsi qu'avec ces nouvelles valeurs, le système mets plus de temps à atteindre l'état d'équilibre. Il faut en effet 100s pour ce cas alors que pour le précédent, on atteignait l'équilibre en moins de 10s. De plus, on peut voir par l'allure du contrôleur, que les valeurs de K, étant plus petites, ont moins d'effet sur le système, ce qui explique que la lenteur de la convergence

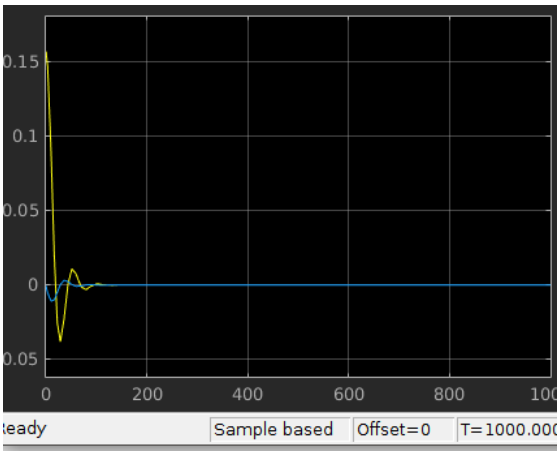
Conclusion Cas 1.2

L'allure de la figure obtenue correspond bien à nos attentes d'un système convergeant vers une position d'équilibre. Les paramètres choisis permettent une bonne visualisation de ce

phénomène. Cependant, on remarque après comparaison avec la courbe précédente, que les valeurs de K ne sont pas optimales pour obtenir une convergence rapide.

< -0.05





t_f

Figure 3 : Cas 1.3

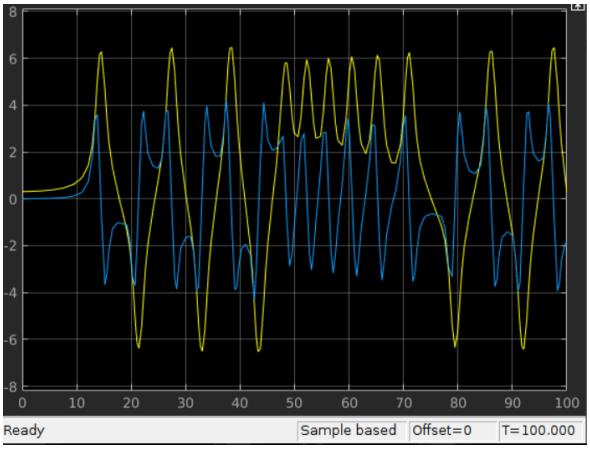
Figure 4 : Cas 1.4

Observations Cas 1.3

On remarque que les conditions initiales pour ce cas sont identiques à celle du *cas 1.2* mise à part du choix de l'intégrateur. L'allure obtenue est très proche de celle obtenu précédemment, cependant, on se rend rapidement compte que les oscillations sont plus marquées ici, étant donné que le deuxième maximum local monte plus que pour le *cas 1.2*. On remarque également que la que la convergence du système est légèrement plus lente dans ce cas, étant donné qu'on rencontre toujours de légères oscillations au bout de 100s.

Observations Cas 1.4

On remarque que les conditions initiales pour ce cas sont identiques à celle du cas 1.3 mise à part le temps t_f. L'allure est identique à celle obtenue précédemment, cependant, le temps t_f choisit permet de mieux visualiser et de s'assurer de la convergence du système. On peut remarquer que dans notre cas, les détails liés à courbe est moins visible en raison d'une échelle plus grande dans l'axe des abscisses. Afin de bien visualiser la courbe et également bien s'assurer de la convergence, il serait donc intéressant de choisir un t_f entre 100s *(cas1.1)* et 1000s *(cas 1.2)*.



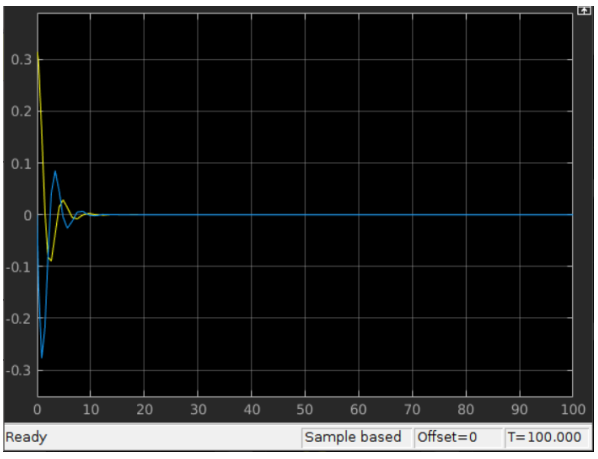


Figure 5 : Cas 1.5

Figure 6 : Cas 1.6

Observation Cas 1.5

Cette fois-ci on a bien k1>g et k2>0 et pourtant le système ne converge pas vers un état d'équilibre.

Cependant, on voit que k2=1 ce qui est proche de 1 et k1=10 ce qui est proche de g. On peut dire qu'on est à la limite des conditions suffisantes de divergence. On remarque également que les paramètres initiaux sont identiques à ceux du cas 1.2 excepté pour la valeur de x_0 dont la première composante est plus grande et donc plus éloigné de la valeur de x à l'équilibre qui est 0. Le système étant initialement dans une zone où l'équilibre est instable, on observe une divergence à la place d'une convergence

Conclusion Cas 1.5

L'allure de la figure est divergente et ne correspond pas au modèle souhaité.

Observations Cas 1.6

On remarque que les conditions initiales sont identiques à celles du *cas 1.5* mise à part pour la valeur de K dont chaque composante est augmentée. On remarque donc bien qu'une augmentation de K permet de forcer la convergence, contrairement à la figure du *cas 1.5*. Le système converge donc malgré une position initiale relativement éloignée de la position d'équilibre. ça force la convergence encore plus, malgré une position initiale instable.

Influence des paramètres

x_0	$K = (k_1 , k_2)$	T_f	*Intégrateur*
Plus il est **proche de la position d'équilibre (0 0)** et plus le système est susceptible de **converger**. Un éloignement trop important pourrait entrainer la divergence du système.	Si **k1>g** et **k2>0** Alors le système converge à condition que x_0 soit assez proche de x_e. Plus **k1** et **k2** sont **grand**, et plus le système est susceptible de **converger**, et plus il est susceptible de converger **rapidement**.	Plus T_f est grand et mieux on arrive à s'assurer que le système converge bien sur le long terme. Plus Tf est petit et mieux on distingue les détails de l'allure de l'évolution du système. Il faut un Tf qui allie les deux conditions citées ci-dessus.	**Euler, ode1** : oscillations plus marquées et pic plus Hauts que pour **Ode45**. Convergence plus lente pour **Euler, ode1**

1.3 Capteurs

Dans cette partie, on introduit un capteur qui renvoie la valeur de $\dot{\alpha}$ et un prédicteur permettant de retrouver la valeur de α à partir de sa dérivée temporelle.
On rajoute donc au schéma Simulink précédent un bloc Capteur et un bloc Prédicteur.

Simulation

Cas	X_0	t_f	$K = (k_1 , k_2)$	pas	Intégrateur
Cas 1	(π/20, 0)	100	(10 ,1)	Par défaut	Ode45
Cas 2	(π/20, 0)	100	(10, 1)	0.001	Euler, ode1
Cas 3	(π/20, 0)	100	(10 ,1)	5	Euler , ode1

Table 2 – Capteur

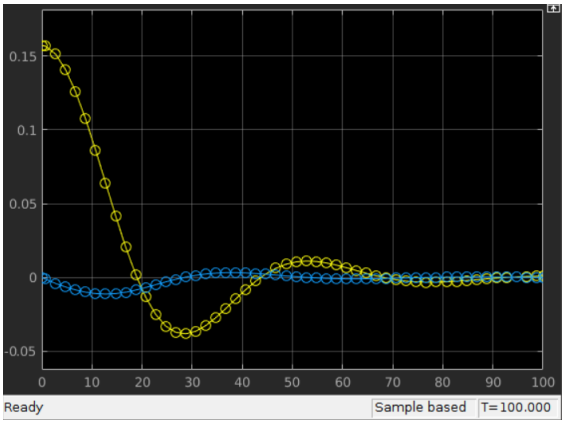


Figure 7 : Cas 2.1

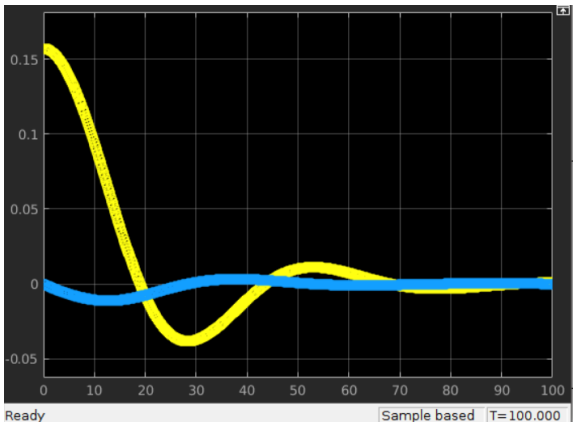


Figure 8 : Cas 2 .2

Observations Cas 2.1

On note que les valeurs initiales sont identiques à celles du *cas 1.2*. Cependant, cette fois-ci, nous avons rajouter une valeur de « pas » par défaut, qui permettra l'échantillonnage. On note que, le choix par défaut de la valeur du pas est acceptable. On ne voit pas beaucoup de discontinuités de la courbe, même si ces derniers sont accentués dans les différents changements de pentes. On arrive également à bien observer un phénomène de convergence du système dans le temps imparti. Après comparaison avec le *cas 1.2*, on note qu'ici les extremums locaux connaissent un pic moins grand que dans le *cas 1.2* qui est par ailleurs plus lisse. Ceci peut être expliqué du fait que notre courbe actuelle ne relève que d'une approximation de alpha.

Observations Cas 2.2

On note que les valeurs initiales sont identiques à celles du *cas 1.3*. Cependant, cette fois-ci, nous avons rajouter une valeur de « pas » relativement petite. Tout comme pour le cas précèdent, les extremums sont moins marqués que pour le *cas 1.3*.
On note qu'avec cette valeur de « pas », on se rapproche de plus en plus au modèle réel, et on limite les erreurs de précision et de discontinuité. Même s'il demeure quelques erreurs notamment liées aux valeurs prises par les extrémums locaux.

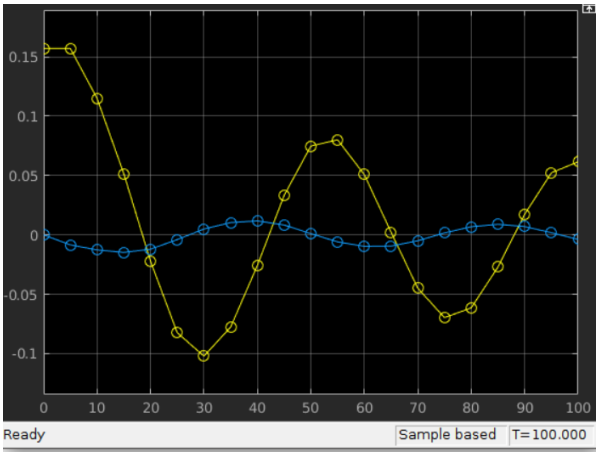


Figure 9 : Cas 2.3

Observations Cas 2.3

On note que les valeurs initiales ici sont identiques à celle du *cas 2* précédent, hormis pour la valeur du pas qui est choisie désormais à 5. Cette valeur est assez élevée, et on arrive à voir une grande discontinuité au niveau de la courbe du système. On ne sait pas réellement ce qui se passe dans l'intervalle séparant deux points, ce qui diminue énormément la précision de l'approximation. D'ailleurs, on remarque que l'extremum local à 30s, atteint -0.1s ce qui est bien éloigné de la réalité où cet extremum est situé à -0.5s. Enfin, avec ce choix de pas, on n'arrive pas à visualiser la convergence du système ce qui nous éloigne du modèle réel.

Influence du « pas » :

Plus le pas choisit est petit est plus l'erreur est minimisée et l'approximation est fidèle à la réalité. Un pas trop élevé ne permet pas de décrire réellement l'évolution du système.

Explication : Equation d'Euler

$\dot{\alpha}_i = \frac{\alpha i+1 - \alpha i}{pas}$ ➔ Dans cette méthode les erreurs se cumulent

II. Simulation du robot Lego

Objectif : Etudier le modèle du robot Lego pendule inversé.

1. Modèle continu

Dans cette partie du TP, on souhaite construire un modèle Simulink continu comportant un système et un contrôleur par retour d'état.

L'état du robot Lego est défini par le vecteur x = (θ Ψ $\dot{\theta}$ $\dot{\Psi}$)

Avec

θ : L'angle moyen des deux roues du robot (en rad).

Ψ : L'angle d'inclinaison du robot (en rad).

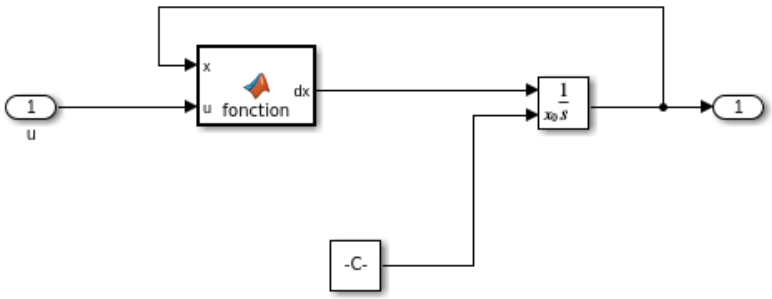
$\dot{\boldsymbol{\theta}}$: La vitesse moyenne des roues du robot (en rad/s).

$\dot{\boldsymbol{\Psi}}$: La vitesse d'inclinaison du robot (en rad/s).

Robot Lego Segway

Ainsi, le modèle Simulink continu comporte le système (le robot Lego) ainsi qu'un contrôleur qui fournit la valeur de u qui représente la tension du moteur à courant continu d'une roue.

Bloc Système



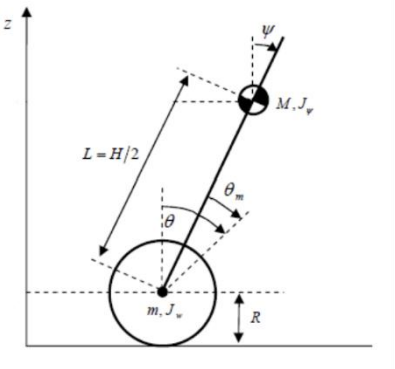


Figure 10 : Bloc Système

Synthèse du contrôleur pour le modèle du robot lego pendule inverse

On commence par le calcul des coefficients de K pour le contrôleur par retour d'état en utilisant les matrices A et B fournies dans l'énoncé, le vecteur des valeurs propres V et la fonction "place" prédéfinie sur Matlab.

Ainsi, avec V = [-136.5905, -2.6555, -3.5026, -5.9946].

On trouve : K = [0.670018499483966, 19.9054724546798, 1.07470992814067, 1.96141961142452]

On remarque que toutes les valeurs propres (éléments du vecteur V) sont strictements négatives ce qui permet de justifier la convergence du système.

Simulation

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/5 0 0)	5	K	Ode45

Table 3 – Données pour le modèle du robot Lego pendule inversé

Résultats et Observations

Légende
__ : θ __ : Ψ
__ : $\dot{\theta}$ __ : $\dot{\Psi}$

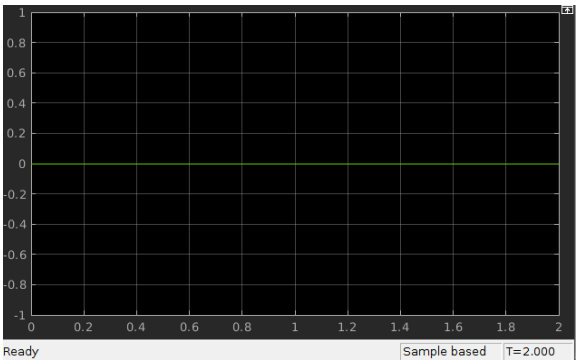


Figure 11 : Cas 1.1 : Etat du système

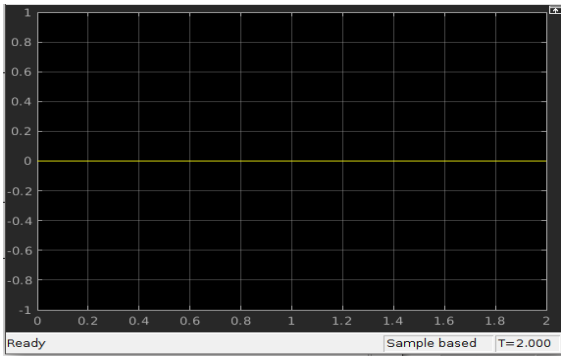


Figure 12 : Cas 1.1 : Contrôleur

Observations Cas 1.1

Dans ce premier cas, la valeur du vecteur décrivant l'état du système x est initialement égale à x_0 = (0 0 0 0) qui correspond à la position d'équilibre x_e. Ainsi, aucune composante de x ne devrait évoluer au fil du temps. Les courbes des variables décrivant l'état du système : θ, Ψ, $\dot{\theta}$ et $\dot{\Psi}$ *(Figure 10)* résultant de la simulation sont constantes et égales à 0, elles s'accordent donc aux prédictions théoriques.
De même, on peut anticiper que la variable u sera constant et égal à 0 au cours du temps car le système ne subit aucune évolution. Autrement dit, il n'y a aucun mouvement ou changement d'état auquel le contrôleur est censé s'opposer, ce qui explique la constance de u à la valeur 0. La courbe des variations au cours du temps de u *(Figure 11)* est donc cohérente.

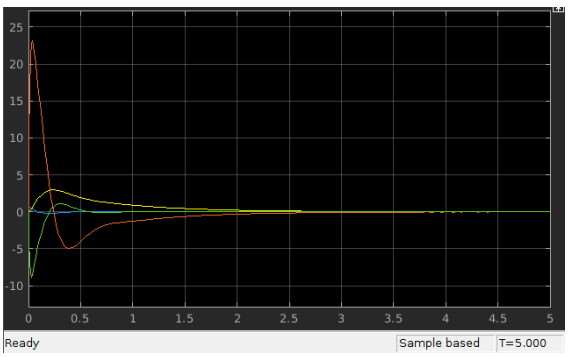


Figure 13 : Cas 1.2 : Etat du système

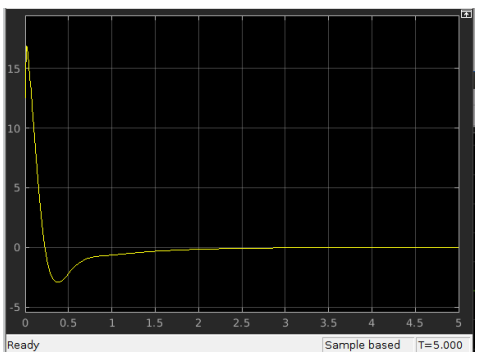


Figure 14 : Cas 1.2 : Contrôleur

Observations Cas 1.2

Dans ce deuxième cas, la valeur du vecteur décrivant l'état du système x est initialement égale à x_0 =(0 π/5 0 0), c'est-à-dire que $\theta_0 = 0$ rad, $\Psi_0 = \pi/.5$ rad et $\dot{\theta}_0$, $\dot{\Psi}_0 = 0$ rad/s.

Etant donné que, Ψ est initialement non nul, le contrôleur essaye d'agir pour stabiliser le système. De ce fait, afin de repasser à 0, la vitesse angulaire $\dot{\Psi}_0$ diminue passant ainsi aux négatifs. Par ailleurs, le contrôleur voulant agir sur le système pour le stabiliser, augmente la valeur de u. Cependant, cette augmentation de u génère par la même occasion une augmentation de l'angle θ et donc également de la vitesse angulaire $\dot{\theta}$ entraînant donc la rotation des roues du robot lui permettant d'avancer. Cependant, pour contrer l'augmentation de θ et $\dot{\theta}$, le contrôleur agit en diminuant la valeur de u, afin de faire converger le système vers sa position d'équilibre x_e. Cette diminution entraîne donc la diminution de la vitesse angulaire $\dot{\theta}$ permettant de replacer θ à une valeur égale à 0. Suite à cela, $\dot{\theta}$ vient se stabiliser à son tour à 0 pour ne plus produire de vitesse angulaire et garder θ dans sa position stabilisée. Ainsi, au bout d'environ 2s, le système arrive à converger vers l'état d'équilibre souhaité pour chacune des composantes de x. Ceci est bien en concordance avec les valeurs propres du coefficient K, qui sont strictement négatives et qui expliquent donc cette convergence.

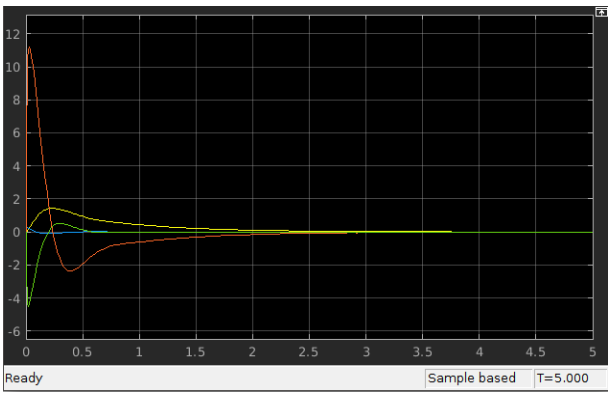


Figure 15 : Cas 1.3 : Etat du système

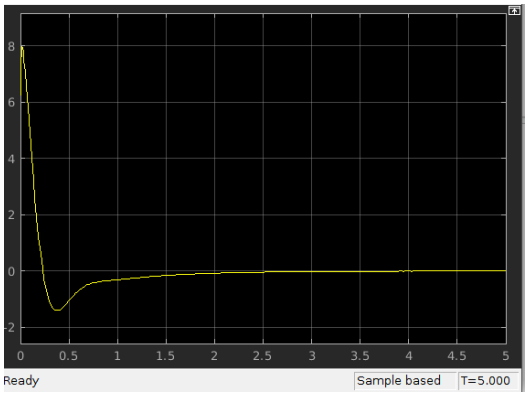


Figure 16 : Cas 1.3 : Contrôleur

Observations Cas 1.3

Dans ce troisième cas, la valeur du vecteur décrivant l'état du système x est initialement égale à x_0 =(0 π/10 0 0).

Etant donné que, Ψ est initialement non nul, le contrôleur essaye d'agir pour stabiliser le système, donc les quatre composantes de x suivront une évolution similaire à celle du cas précédent *(Cas 1.2).* Cependant, il est important de noter que la valeur initiale de Ψ est égale à la moitié de celle du cas précédent (pi/10=(pi/5) /2), la rapprochant ainsi de la valeur d'équilibre. Par conséquent, le contrôleur aura une influence moindre sur le système par rapport au cas précédent. Cela explique pourquoi les amplitudes des pics des courbes des différentes variables sont maintenant divisées par 2 par rapport au *Cas 1.2.*

2. Introduction des capteurs et actionneurs

L'état du système est observé par des capteurs qui ne restituent pas l'intégralité des composantes de l'état. Un gyroscope mesure la vitesse de changement d'angle du corps du robot) et un capteur mesure l'angle θ(t). Pour compenser cette perte d'information, il faut introduire un sous-système prédicteur qui recalcule les informations manquantes à partir des informations disponibles.

Simulation

On utilise les mêmes conditions initiales que pour la table 3.

Résultats et Observations

Figure 17 : Cas 2.1 : Etat du système

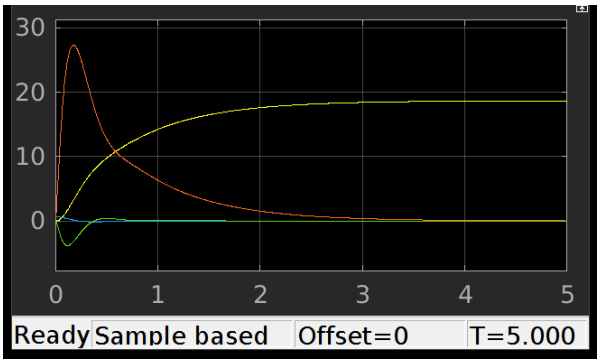


Figure 18 : Cas 2.2 : Contrôleur

Observations Cas 2.1

Dans ce premier cas, le système est initialement dans sa position d'équilibre $x_0 = (0\ 0\ 0\ 0) = x_e$. Par conséquent, il n'y aura aucune évolution au cours du temps de x. Ainsi, les courbes de θ, Ψ, $\dot{\theta}$, $\dot{\Psi}$ restent constantes égales à 0. Et de même pour la courbe de u car le contrôleur n'agit pas sur le système s'il est en position d'équilibre.

Observations Cas 2.2

Un raisonnement analogue au précédent permet d'expliquer l'évolution de Ψ et $\dot{\Psi}$.
On remarque que les conditions initiales choisies sont identiques à celles du cas précédent mise à part pour la valeur x_0 qui est désormais plus proche de la position d'équilibre x_e.
Ainsi, on observe que, une fois de plus, suite à l'action du contrôleur qui veut agir pour replacer Ψ à 0, $\dot{\theta}$ augmente ce qui entraîne l'angle la déviation vers les positifs de θ.
De ce fait, lorsque θ augmente, les roues tournent dans un seul sens et le robot avance jusqu'à ce que θ et Ψ deviennent constants et donc que le robot se stabilise.
Par la suite, $\dot{\theta}$ diminue jusqu'à annulation totale de cette dernière, ce qui stabilise θ à une valeur non nulle.

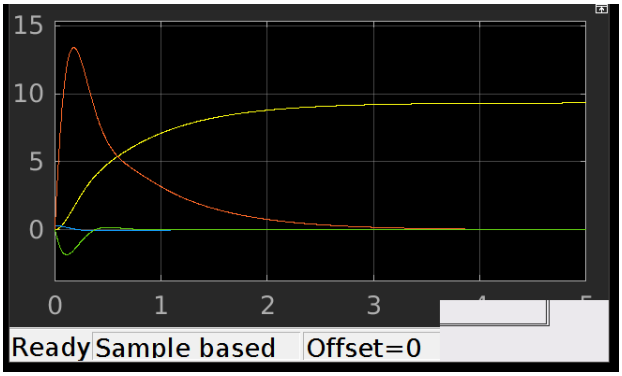


Figure 19 : Cas 2.3

Observations Cas 2.3

Dans ce troisième cas, la valeur du vecteur décrivant l'état du système x est initialement égale à $x_0 = [0\ \pi/10\ 0\ 0]$. Etant donné que, Ψ est initialement non nul, le contrôleur essaye d'agir pour stabiliser le système, donc les quatres composantes de x suivront une évolution similaire à celle du cas précédent *(Cas 1.2)*. Cependant, il est important de noter que la valeur initiale de Ψ est égale à la moitié de celle du cas précédent (pi/10=(pi/5)/2), la rapprochant ainsi de la valeur d'équilibre. Par conséquent, le contrôleur exercera une influence moindre sur le système par rapport au cas précédent. Cela explique pourquoi les amplitudes des pics de courbes des différentes variables sont maintenant quasiment divisées par 2 précédent.

3.Construction du modèle hybride

Etape 1 : On introduit dans le capteur un bloc Zero_Order Hold.
⇒ L'état reconstruit en sortie du capteur est ainsi discret
Etape 2 : Modification du prédicteur pour utiliser des opérateurs discrets

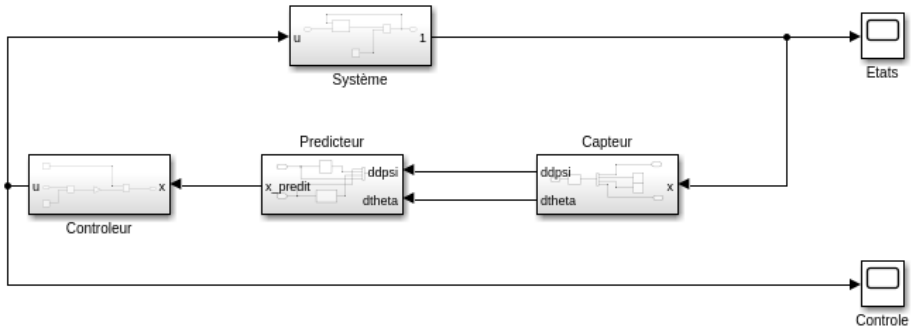


Figure 20 : Nouveau schéma Simulink

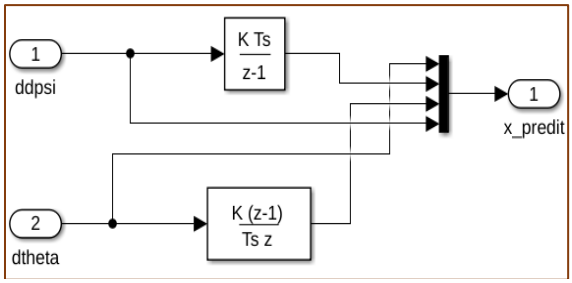


Figure 21 : Bloc Prédicteur

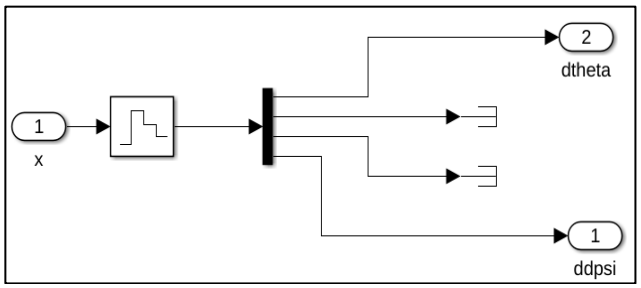


Figure 22 : Bloc Capteur

Simulation

On utilise les mêmes conditions initiales que pour la table 3.

Résultats et Observations

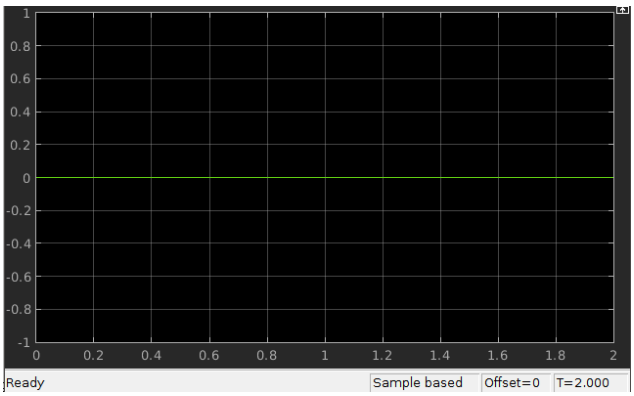


Figure 23 : Cas 3.1

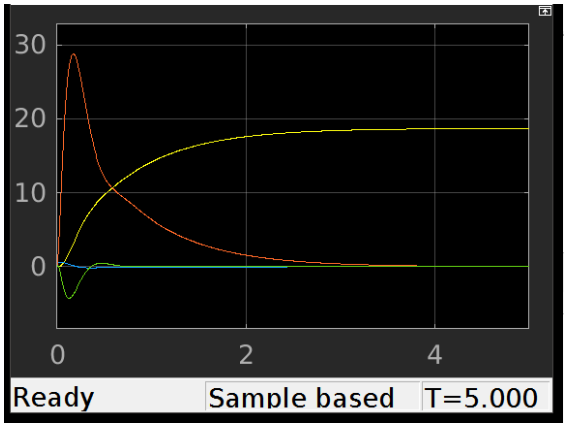


Figure 24 : Cas 3.2

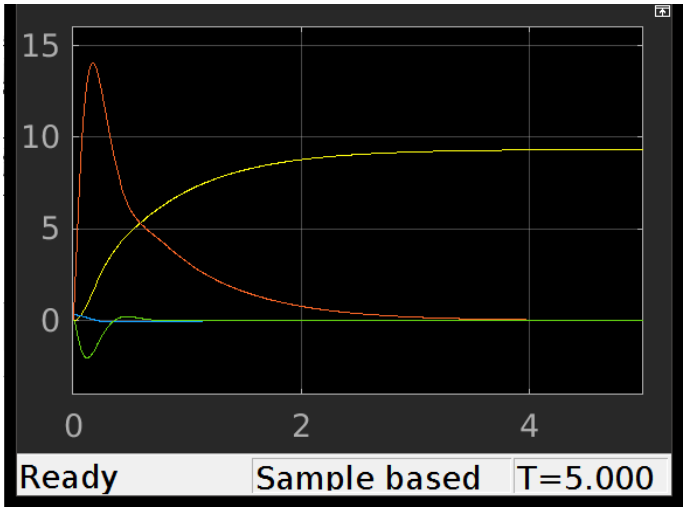
Observations Cas 3.1

Dans ce premier cas, le système est initialement dans sa position d'équilibre $x_0 = (0\ 0\ 0\ 0) = x_e$. Par conséquent, il n'y aura aucune évolution au cours du temps de x. Ainsi, les courbes de θ, Ψ, $\dot{\theta}$, $\dot{\Psi}$ restent constantes égales à 0. Et de même pour la courbe de u car le contrôleur n'agit pas sur le système s'il est en position d'équilibre.

Observations Cas 3.2

On remarque que ce cas possède les mêmes conditions initiales que le cas 2 de la partie précédente. Ceci explique par ailleurs, les fortes ressemblances au niveau des courbes obtenues.

L’analyse de l’évolution du système se fait donc de manière analogue. Cependant, on remarque quelques légères différences notamment au niveau des extremums locaux, qui semblent être plus accentués dans le cas discret que dans le cas continu.



Observations Cas 3.3
Dans ce troisième cas, la valeur du vecteur décrivant l’état du système x est initialement égale à x_0 = (0 π/10 0 0).
Etant donné que, Ψ est initialement non nul, le contrôleur essaye d’agir pour stabiliser le système, donc les quatre composantes de x suivront une évolution similaire à celle du cas précédent *(Cas 1.2)*. Cependant, il est important de noter que la valeur initiale de Ψ est égale à la moitié de celle du cas précédent (pi/10=(pi/5)/2), la rapprochant ainsi de la valeur d’équilibre. Par conséquent, le contrôleur aura une influence moindre sur le système par rapport au cas précédent. Cela explique pourquoi les amplitudes des pics de courbes des différentes variables sont maintenant quasiment divisées par 2 précédent.

Conclusion

En conclusion, lors de ces TP, les simulations du pendule inversé et du robot Lego ont permis de nous assurer de l’efficacité du contrôle par retour d’état pour stabiliser ces systèmes dynamiques.

Généralement, les résultats obtenus lors de ces séances, nous ont permis d’observer un phénomène de convergence assez rapide ce qui démontre la robustesse du contrôleur dans des conditions initiales variées, ouvrant ainsi la voie à des applications pratique en ce qui concerne la régulation automatique vu lors du TP4.