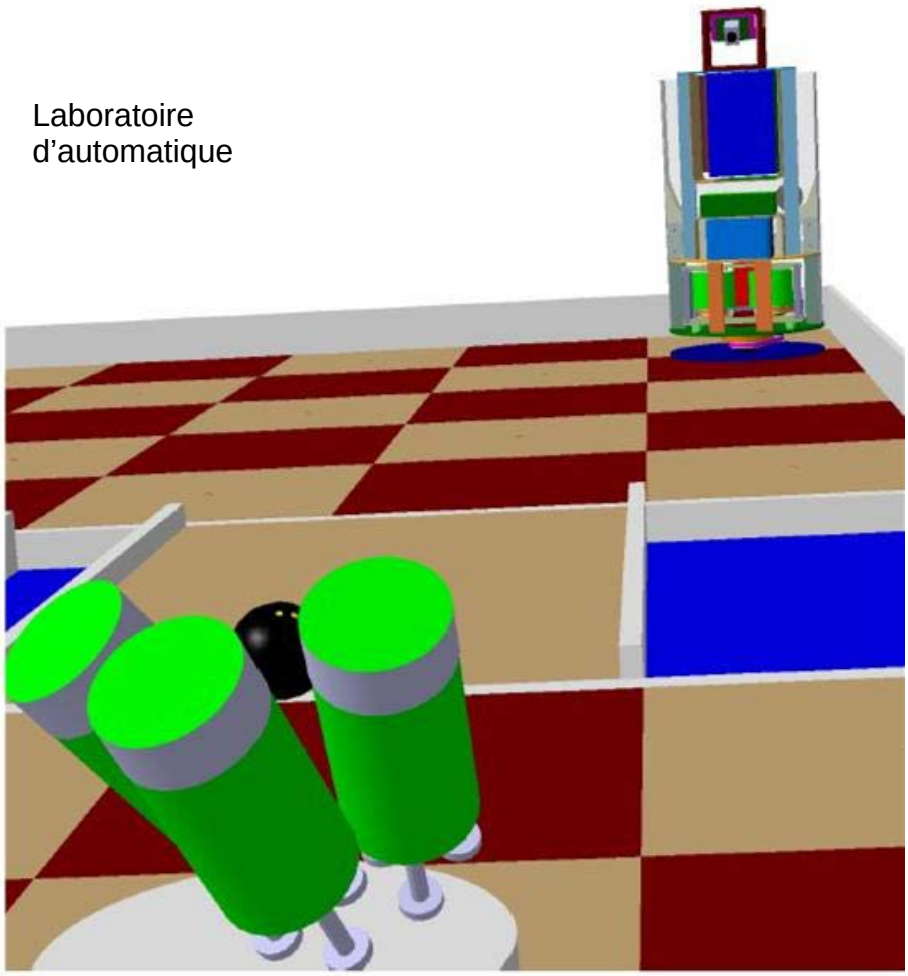
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Commande de la tourelle d'un robot

Laboratoire
d'automatique



Projet de 6ème semestre
Génie Mécanique

Sylvain Rudaz

Assistant : Christophe Salzmann
Professeur : Denis Gillet

23 juin 2005

Table des matières

1. Introduction 3
1.1. Le contexte 3
1.2. Objectifs et vue d'ensemble 5
1.3. Système physique 6
1.4. Cahier des charges 7
2. Modélisation 8
3. Expérimentation 11
3.1. Coefficients de Ziegler-Nichols théoriques 11
3.2. Coefficients de Ziegler-Nichols expérimentaux 12
3.3. Idées infructueuses 13
3.4. Antireset windup 14
3.5. Filtrage du terme dérivateur 14
3.6. Consignes de vitesse sous forme incrémentale 14
3.7. Estimation de la perturbation 17
4. Conclusions 20
4.1. Choix du régulateur et remarques 20
4.2. Perspectives 20
Annexe 21
Bibliographie 23

Projet de semestre
Commande de la tourelle d'un robot

1. Introduction

1.1. Le contexte

Concours de robotique

La compétition de robotique autonome SwissEurobot existe depuis 1998. Elle permet aux équipes suisses de se qualifier pour la finale européenne Eurobot. En 2005, les deux manifestations ont eu lieu les 19, 20 et 21 mai à Yverdon-les-Bains.

L'équipe Team-ID dont je fais partie, a participé pour la 3ème année consécutive a ce concours. Elle est composée de 9 étudiants de quatre sections différentes de l'EPFL et d'un étudiant de l'ETML, chargé principalement de l'usinage des pièces dans le cadre de son année d'approfondissement.

Le règlement change d'une année à l'autre mais la philosophie reste la même. Le ou les robots de 2 équipes s'affrontent dans un match sous forme d'un jeu basé sur le fair-play.

Le règlement 2005

Voici un résumé du règlement :
« Cette année, les robots jouent au bowling.
Chaque équipe doit construire un ou deux robots. Les matchs se jouent entre deux équipes et durent une minute trente.
Une couleur de quilles est attribuée à chaque équipe.
Pour gagner la partie, les robots doivent renverser leurs quilles. Au départ, celles-ci se trouvent de l'autre côté d'un fossé. Les robots peuvent aussi relever les quilles, de la couleur adverse, renversées par l'autre équipe.
L'équipe qui a le plus de quilles de sa couleur renversées en fin de match est déclarée vainqueur. »

Chaque équipe reçoit à la fin du match :

- un point par quille renversée si elle est de la couleur qui lui a été attribuée,
- un point pour deux quilles de l'autre couleur debout avec les vis en l'air.

De plus, l'équipe se voit attribuer 3 points additionnels par match, en cas de victoire.

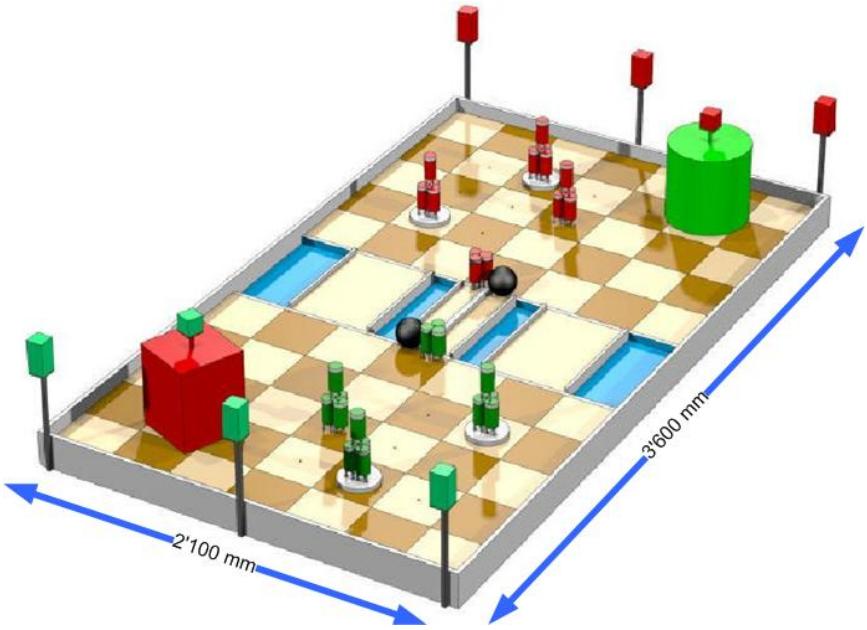


Figure 1 : terrain de jeu

Le robot

La stratégie de l'équipe est d'avoir deux robots. Un défenseur relevant ses propres quilles renversées par l'adversaire et un attaquant. Le but de ce dernier est de renverser les quilles situées de l'autre côté du terrain. C'est la tâche la plus importante.

L'attaquant est un robot immobile et peut faire tomber les quilles à distance en lançant les huit balles de squash embarquées.

Un système de tir constitué de deux rouleaux de relativement grande inertie tournant à haute vitesse lance les balles en direction de la cible choisie à la manière d'un lanceur de balles de tennis.

Les balles sont préalablement stockées dans un réservoir et sont distribuées aux rouleaux grâce au système d'amenée.

Un système de repérage, équipé d'un télémètre laser aligné avec le système de tir, permet de déterminer la position des quilles isolées ou des tas de quilles.

Le tout est placé sur une tourelle orientable.

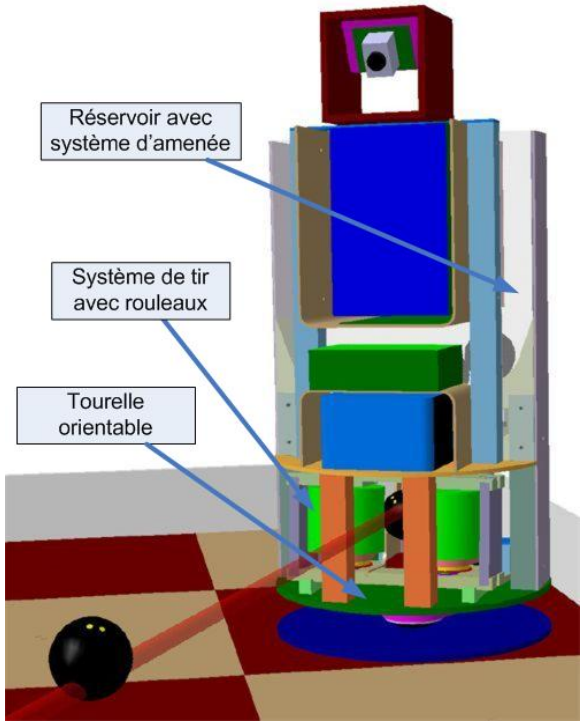


Figure 2 : Robot attaquant avec ses systèmes

Avant le coup d'envoi, ce robot profite de sa tourelle pour scanner l'environnement et mémoriser l'emplacement des tas de quilles. Selon le règlement, s'il avait été monté sur roues, il n'aurait pas pu tourner avant le départ du match.

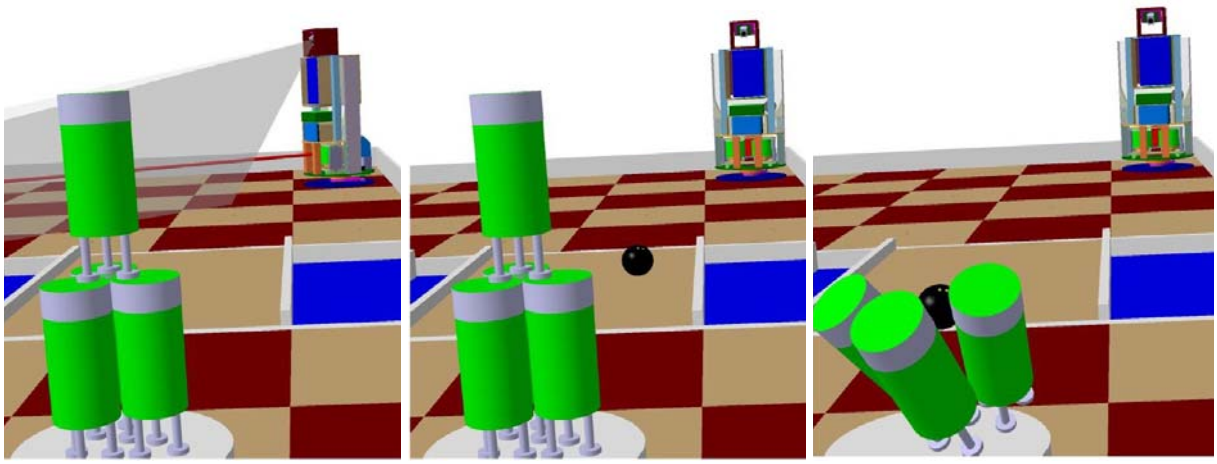


Figure 3 : Le robot attaquant en action

Durant les dix premières secondes après le départ, il renverse les quatre tas de quilles dont il a mémorisé l'emplacement, avec le nombre de balles nécessaires. Il tire les balles restantes sur les quilles relevées pendant le reste du match qui dure 90 secondes.

	Tâches	Moyens	Actionneurs	Transmissions	Capteurs
1	Repérer	Segmentation dynamique	-	-	Caméra et laser
2	Viser	Tourelle orientable	Moteur DC	Courroie	Encodeur
3	Amener les balles	Rouleau	Moteur DC	Directe	Barrière optique
4	Tirer	Deux rouleaux	Deux moteurs DC	Courroies	Encodeurs

Tableau 1 : Récapitulatif des tâches du robot attaquant

1.2. Objectifs et vue d'ensemble

L'objectif principal de ce projet est la régulation de la tourelle orientable du robot attaquant décrit précédemment.

Objectifs techniques :

- garantir un comportement correct en asservissement et régulation de position angulaire et de la vitesse,
- satisfaire le cahier des charges fixé par le règlement et par l'équipe.

Ce projet s'inscrivant au 6e semestre, il comporte également d'autres objectifs :

- comprendre et avoir une certaine intuition de la problématique de la commande d'un système réel,
- sentir les limitations de la commande PID,
- collaboration avec des personnes venant d'autres filières d'études (pluridisciplinarité).

1.3. Système physique

Tout le robot tourne sur un axe central de 20 mm, muni d'un gros roulement à billes. Un moteur fixé sur la tourelle fait tourner une roue dentée reliée à celle fixée à la plaque au sol, par une courroie crantée.

Pour assurer une mesure précise de l'angle de la tourelle, un encodeur est placé directement sur l'axe principal, et donc en sortie. Comme la position n'est pas mesurée *via* un encodeur moteur, elle est moins sujette à des erreurs.

Un système de rattrapage de jeu a été mis en place pour annuler le jeu de 2° à la sortie du réducteur et celui de la courroie, malgré un facteur de réduction réduisant le jeu à un demi degré et une tension de courroie adéquate. Ceci se fait à l'aide d'un ressort de traction (non représenté dans la figure ci-dessous) mis sous tension par l'intermédiaire d'un fil enroulé à l'extrémité inférieure de l'axe, juste au dessus de la plaque de base fixe par rapport au sol.

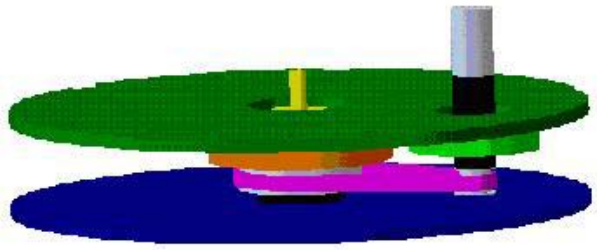


Figure 4 : Tourelle

Voici les caractéristiques principales du robot :

- Diamètre de 20 cm, hauteur de 40 cm, masse d'environ 10 kg.
- Un moteur DC de 23W, équipé d'un encodeur magnétique (non utilisé).
- Encodeur optique à 2 canaux, 500 impulsions par tour, monté directement sur l'axe de la tourelle.
- Système embarqué : carte rokEPXA équipée d'un processeur « hardcore » ARM 133 Mhz et d'un processeur « softcore » NIOS 32 MHz. La régulation se fera sur le NIOS.

Ce robot a été entièrement conçu sur CATIA®, logiciel de Dessin Assisté par Ordinateur.

1.4. Cahier des charges

Le système mécanique donné ne peut être modifié. Le système informatique ainsi que toute la partie électronique sont également imposés. La régulation sera faite sur le NIOS, qui ne travaille qu'avec des **nombres entiers**.

Le capteur incrémental fixé directement en sortie d'axe comporte 2000 incréments par tours. Chaque incrément représente ainsi 0.18° d'angle. Sachant qu'il faudra toucher une cible à plus ou moins 2 [cm] autour de la position théorique (en largeur) et ceci à une distance d'environ 3.5 [m], on doit donc avoir une précision angulaire d'environ 0.33° ce qui représente 1.8 incrément. On ne tolère donc qu'un **écart angulaire de 1 incrément**. En effet, le système de tir présente également une légère dispersion dans le même ordre de grandeur.

Le robot a une autre particularité qui est l'asymétrie due à la présence du ressort pour le rattrapage de jeu. Notons que l'introduction d'un deuxième ressort dans le système mécanique n'aurait pas été un choix judicieux puisqu'il y aurait alors deux ressorts en parallèle, et le jeu qu'il s'agit d'annuler serait toujours présent.

Le déroulement d'un match selon la stratégie de l'équipe (hors projet) se fait comme suit : on demande au robot de faire un scanning de la table de jeu pour déterminer l'emplacement des quilles. Puis, lorsqu'une cible a été détectée, le robot doit atteindre la position souhaitée quasi-instantanément mais dans tous les cas en **moins d'une demie seconde**.

Si le robot ne travaille pas en recherche dynamique, c'est-à-dire qu'il mémorise les positions pour n'y aller qu'ultérieurement, la position de consigne doit être atteinte également dans un temps de l'ordre de la seconde depuis n'importe quelle autre position.

Figure 5 : Résumé du projet

Contrainte temporelle : le robot doit de plus être opérationnel le 19 mai 2005, date du concours. Le projet de semestre commençant officiellement le 7 mars 2005, cela laisse théoriquement une période de **10 semaines** dans le calendrier scolaire pour mener à terme le projet.

2. Modélisation

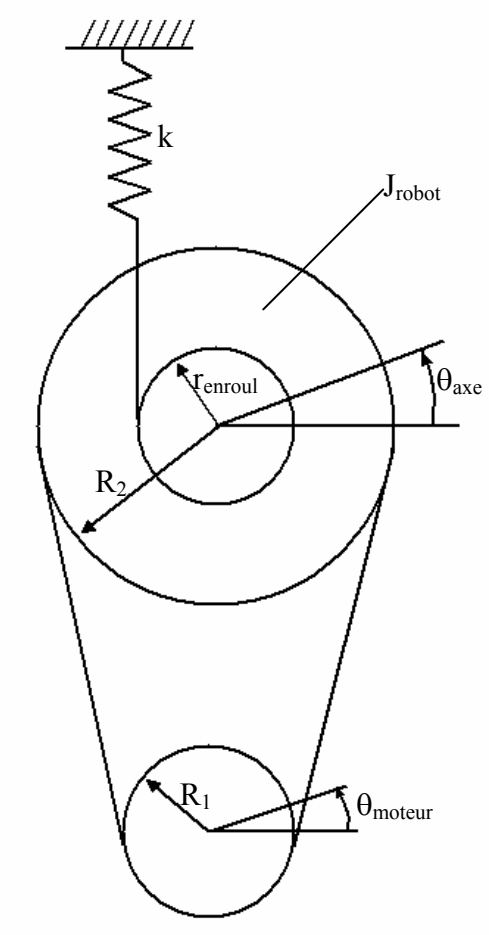


Figure 6 : modèle mathématique

Données :

Inertie du robot : $J_{robot} = 0.04\ [kg \cdot m^2]$
(calculé à l'aide du logiciel DAO)

Rapport de réduction : $n = \frac{R_2}{R_1} = \frac{56}{1}$
(donné par les fabricants de poulies et moteur)

Rayon d'enroulement : $r_{enroul} = 0.017\ [m]$
(donné par le fabricant de la poulie)

Constante moteur : $K \cdot \Phi = 0.0346\ \left[\frac{Nm}{A}\right]$
(donné par le fabricant du moteur)

Résistance du circuit : $R = 5.78\ [\Omega]$
(donné par le fabricant du moteur)

Résistance du ressort : $k_{ressort} = 158\ \left[\frac{N}{m}\right]$
(mesuré dans le domaine linéaire)

Hypothèses :

- courroie infiniment rigide
- pas de pertes par frottements visqueux ni sec

Pour un moteur à courant continu et à excitation séparée, nous connaissons la relation :

$$u(t) = R \cdot i(t) + K \cdot \Phi \cdot \dot{\theta}_{moteur}(t) + L\frac{di(t)}{dt} \qquad (1)$$

qui se simplifie en négligeant l'inductance du circuit :

$$u(t) = R \cdot i(t) + K \cdot \Phi \cdot \dot{\theta}_{moteur}(t)\ . \qquad (2)$$

De plus, nous savons que le couple moteur est directement proportionnel au courant :

$$M_{moteur}(t) = K \cdot \Phi \cdot i(t) \qquad (3)$$

Nous avons également la relation entre le couple moteur est le couple moteur ramené à l'axe :

$$n = \frac{R_2}{R_1} = \frac{\theta_{moteur}}{\theta_{axe}} = \frac{\dot{\theta}_{moteur}}{\dot{\theta}_{axe}} = \frac{\ddot{\theta}_{moteur}}{\ddot{\theta}_{axe}} = \frac{M_{moteur,axe}}{M_{moteur}} \tag{4}$$

En ce qui concerne le ressort, nous supposons qu'il a une caractéristique force-déplacement affine. La force à déplacement nul est une force de précontrainte, si bien que :

$$F_{ressort} = F_0 + k \cdot \theta_{axe} \cdot r_{enroul}$$

et donc :

$$M_{ressort} = -r_{enroul} \cdot (F_0 + k \cdot \theta_{axe} \cdot r_{enroul}) \tag{5}$$

Nous pouvons ensuite énoncer la 2ème loi de Newton selon un axe sortant du plan de la feuille et passant par l'axe du robot :

$$\sum M_{<axe>} = -r_{enroul} \cdot (F_0 + k \cdot \theta_{axe} \cdot r_{enroul}) + n \cdot M_{moteur} = J_{robot} \cdot \ddot{\theta}_{axe} \tag{6}$$

En introduisant les différents moments rapportés au moteur, nous obtenons l'équation dynamique :

$$J_{robot} \cdot R \cdot \ddot{\theta}_{axe}(t) + (K\Phi)^2 \cdot n^2 \cdot \dot{\theta}_{axe}(t) + k_{ressort} \cdot R \cdot r_{enroul}^2 \; \theta_{axe}(t) = K\Phi \cdot n \cdot u(t) - F_0 \cdot R \cdot r_{enroul}$$

On trouve donc la fonction de transfert analogique du système en boucle ouverte pour une régulation en position :

$$G(s) = \frac{\Theta(s)}{U(s)} = \frac{K\Phi \cdot n}{J_{robot} \cdot R \cdot s^2 + (K\Phi)^2 \cdot n^2 \cdot s + R \cdot k_{ressort} \cdot r_{enroul}^2}$$

En introduisant les différentes valeurs, on peut trouver les pôles du système en boucle ouverte :

$$p_1 = -16.17$$

$$p_2 = -0.07$$

La BIBO stabilité en boucle ouverte est garantie ce qui semble étonnant à première vue. Ceci provient de la contribution du ressort qui devient importante pour de grandes valeurs de θ.

On peut calculer la réponse du système à un saut unité. On trouve :

$$\theta(t) = 1.9376 \cdot (3.7889 + 0.0166 \cdot e^{-16.1677 \cdot t} - 3.8056 \cdot e^{-0.0706 \cdot t})$$

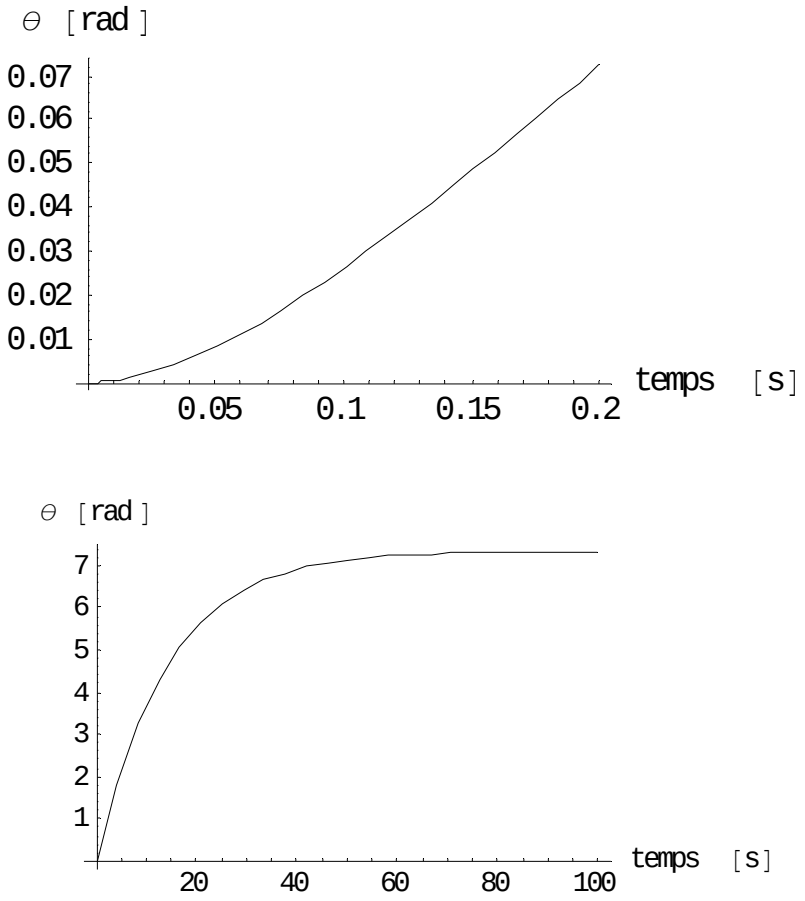


Figure 7 : réponse indicielle du système

De ce résultat théorique, on peut trouver le point d'inflexion, dessiner la tangente et y calculer la pente. On détermine ainsi un des deux coefficients de la première méthode de Ziegler-Nichols. Le deuxième coefficient est obtenu en regardant l'abscisse à l'origine de la tangente.

$a = 0.5061$ et $L = 0.6173$

Grâce à ces deux valeurs, on trouve les coefficients théoriques des régulateurs P, PI et PID :

	Kp	Ti	Td
P	3.2002	inf.	0
PI	2.8804	2.0371	0
PID	3.8406	1.2346	0.3087

3. Expérimentation

3.1. Coefficients de Ziegler-Nichols théoriques

La première tâche avant l'implémentation consiste à changer les unités. La tension n'est plus en volts mais en PWM. On a que 24 V correspondent à 255 de PWM.

Les angles sont convertis comme suit : 2π radians correspondent à 2000 incréments.

En insérant les coefficients trouvés de manière théorique dans notre algorithme de commande numérique, nous obtenons pour un saut de 100 incréments et pour un régulateur PID :

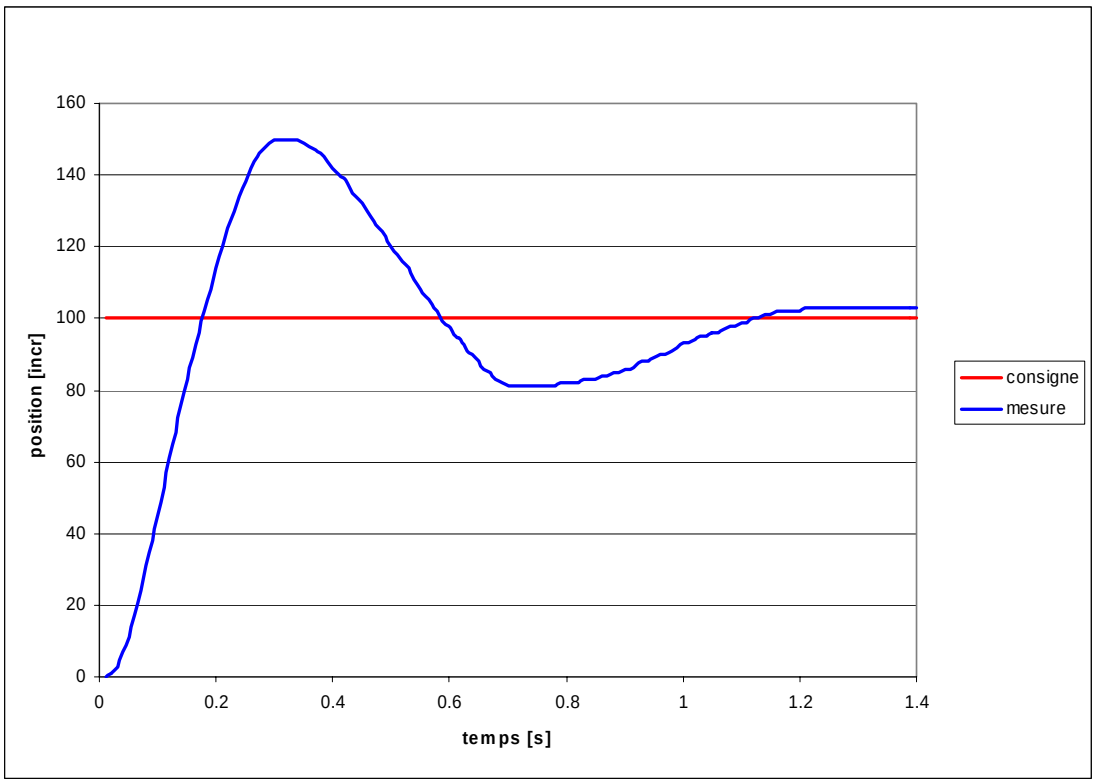


Figure 8 : réponse à un saut de consigne de 100, avec PID théorique (système réel)

Commentaires

On remarque que le comportement n'est de loin pas optimal :

- large dépassement de la valeur de consigne,
- mauvais amortissement du régime transitoire,
- statisme non négligeable même après une seconde et demie.

Les coefficients théoriques sont apparemment mal adaptés. De plus, si le ressort n'est pas relié, le modèle est franchement mauvais. Ceci s'explique par le fait que le ressort augmente la rigidité du système et enlève le jeu. Les divers frottements sont donc masqués.

3.2. Coefficients de Ziegler-Nichols expérimentaux

L'idée est de comparer le comportement en boucle ouverte par le même type de régulateur mais avec des coefficients déterminés, cette fois-ci, expérimentalement.

Tout d'abord, il faut remarquer que la tension initiale du ressort joue un rôle important dans la dynamique du système.

Ci-dessous, plusieurs essais permettent de visualiser ce phénomène.

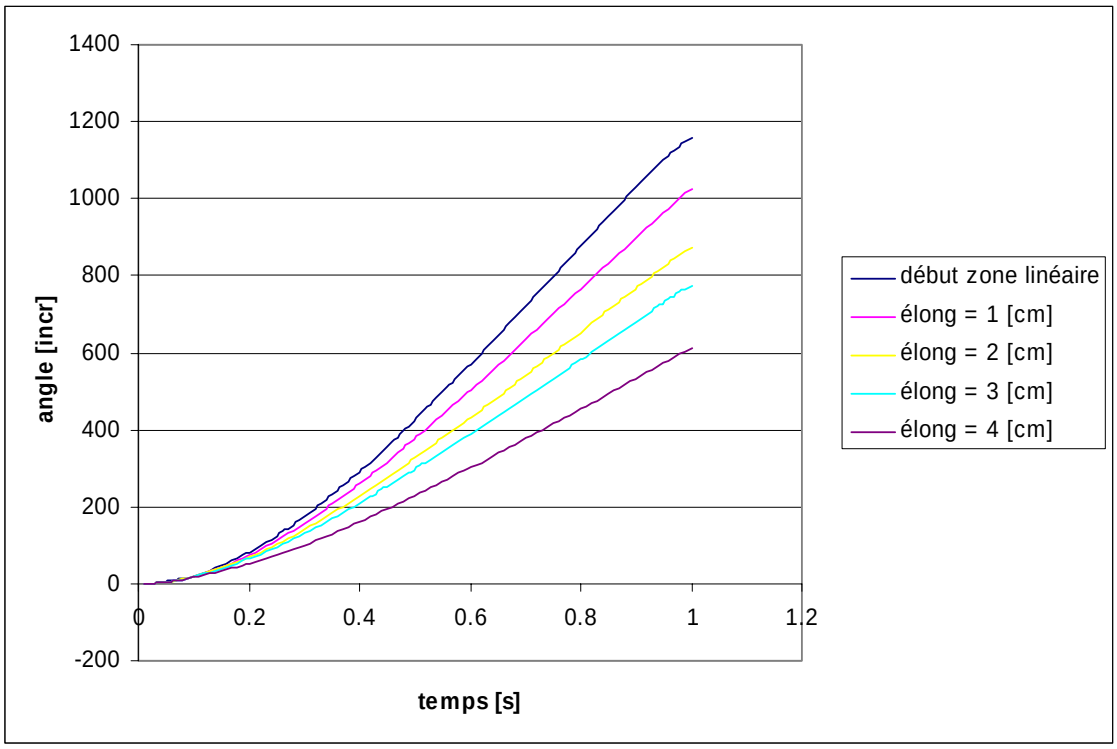


Figure 9 : réponse à un saut de 150 PWM en BO (système réel), diverses tensions ressort

Il s'agit donc de fixer une fois pour toute, l'élongation initiale. Celle-ci a été arrêtée à 4 [cm]. On augmente ainsi la rigidité comme énoncé dans le paragraphe précédent, ce qui est bénéfique pour la régulation.

Par la première méthode de Ziegler-NIchols à nouveau, les coefficients du PID sont déterminés :

$$Kp = 6$$
$$Ti = 0.16$$
$$Td = 0.08$$

On peut montrer le comportement en boucle fermée avec le régulateur PID qui emploie ces coefficients.

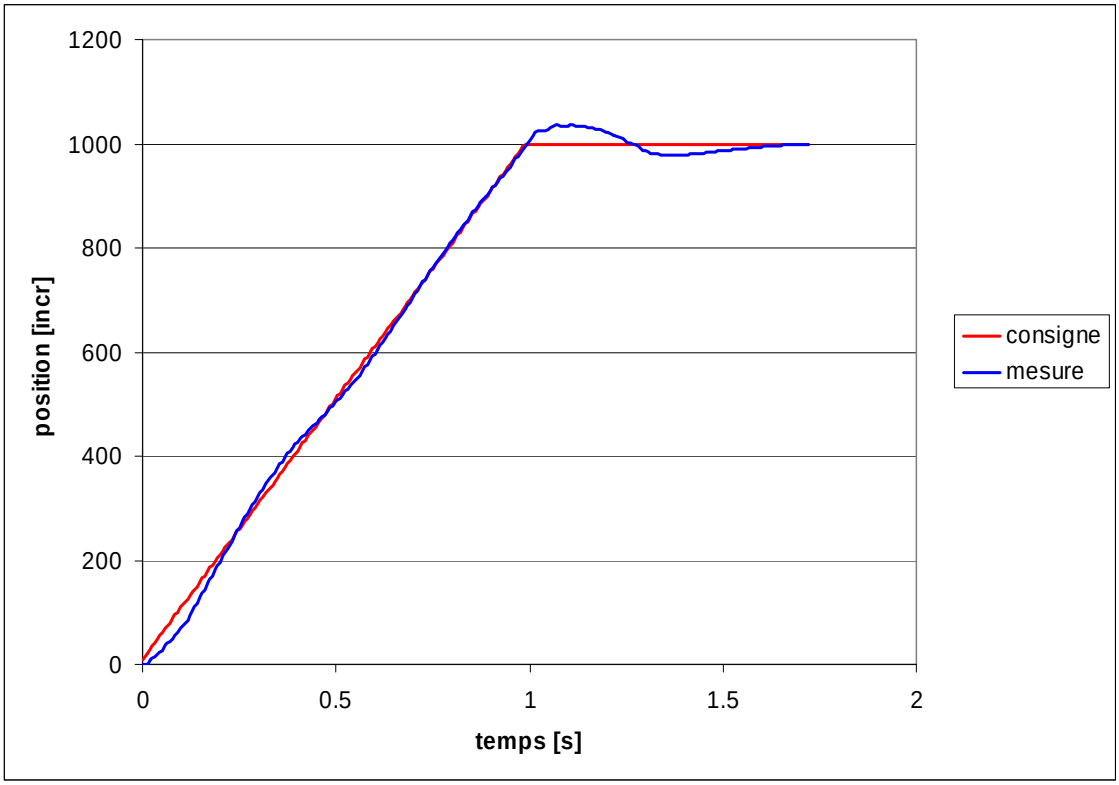


Figure 10 : réponse à une rampe, avec PID expérimental (système réel)

Commentaires

Le résultat est étonnamment convenable, mais pas excellent. Il y a toujours un comportement oscillatoire, un dépassement de la valeur de consigne et un mauvais amortissement du régime transitoire.

Difficultés

Il est difficile, à ce stade, de trouver des coefficients convenables simplement par l'essai de valeurs. En effet, ils sont tous dépendance les uns des autres.

Démarche

Deux possibilités se présentent pour essayer d'améliorer le comportement en boucle fermée :

- essayer de trouver des astuces et continuer une synthèse purement expérimentale
- synthétiser un régulateur avec une des méthodes plus élaborées, tirées de la littérature spécialisée.

Vu le stade auquel était le projet au moment de cette décision, la première solution a été retenue pour plusieurs raisons. La principale était la date du concours qui était toute proche. Une autre, le niveau de connaissance à ce moment-là. De plus, une synthèse par exemple dans les diagrammes de Bode auraient demandé un grand investissement sur le système réel.

3.3. Idées infructueuses

Plusieurs approches ont été testées sans toutefois pouvoir aboutir à un résultat correct :

a) commande a priori
b) changement de la forme de la consigne pour mieux répondre à la dynamique du système,
c) deux régulateurs différents selon le sens de rotation.

L'abandon de ces trois concepts s'explique facilement même qualitativement. L'idée de la commande à priori semble difficile à mettre en œuvre. La tension du ressort n'est pas identique pour toute plage de position angulaire de la tourelle.

Changer la forme de la consigne serait plus difficile à mettre en œuvre et à générer. De plus, la puissance de calcul ne permettrait pas de faire une multitude de calculs complexes. L'emploi de fonctions trigonométriques nous aurait obligé de créer des tables puisque le NIOS ne fonctionne qu'avec des entiers.

La troisième idée quant à elle serait une solution peu élégante et augmenterait inutilement la difficulté puisqu'il y aurait deux régulateurs à synthétiser.

3.4. Antireset windup

Pour pallier au comportement connu et gênant de l'emballement du terme intégral, un antireset windup (ARW) a été implanté. La méthode employée consiste simplement à saturer ce terme. Il s'agit d'une composante non linéaire.

A nouveau, la synthèse de cet élément s'est faite expérimentalement. Sur tous les tests effectués, nous avons remarqué que le terme intégral était rarement en saturation même avec des conditions de bornes strictes.

Le déroulement d'un match est relativement prédictible. Comme le robot a la forme d'un cylindre, cet ARW est donc plus une sécurité en cas de blocage par un robot adverse, relativement peu probable.

3.5. Filtrage du terme dérivateur

Le comportement avec un filtre basique sur le terme dérivateur étant bizarrement moins bon que sans celui-ci, il n'a pas été introduit. D'autre filtres n'ont pas été testés.

3.6. Consignes de vitesse sous forme incrémentale

En observant la figure 10, on remarque que suivre une consigne en forme de rampe sans traînée semble faisable. Pour pouvoir garantir ceci, on peut déjà essayer de borner la pente de la rampe. Ci-dessous, 3 réponses pour différentes pentes maximales.

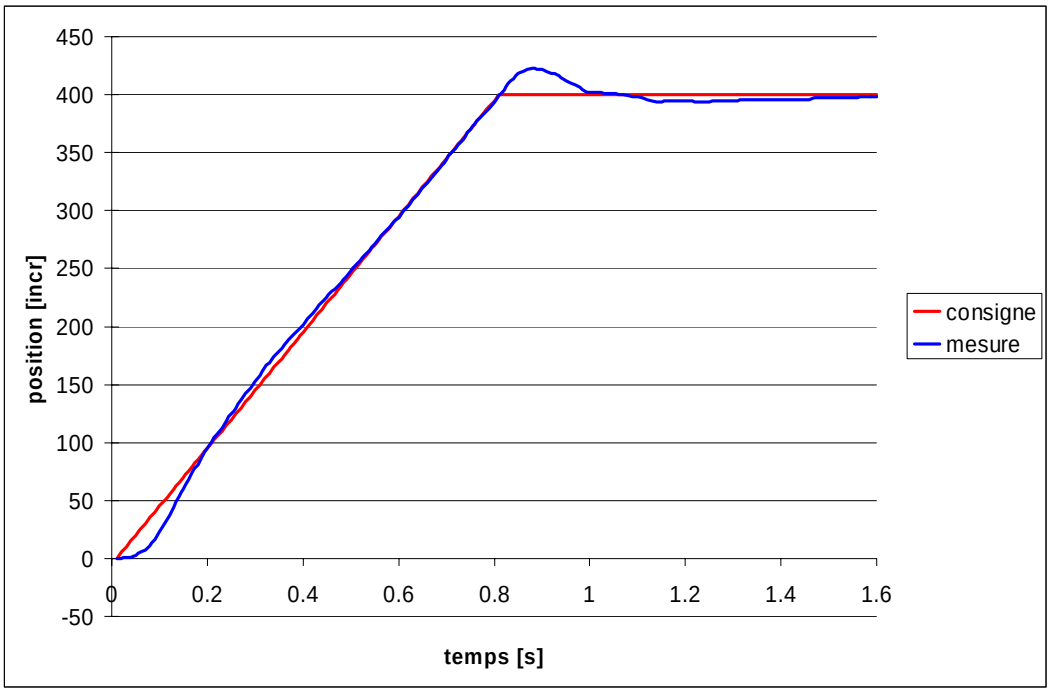


Figure 11 : rampe avec PID expérimental, pente max = 5

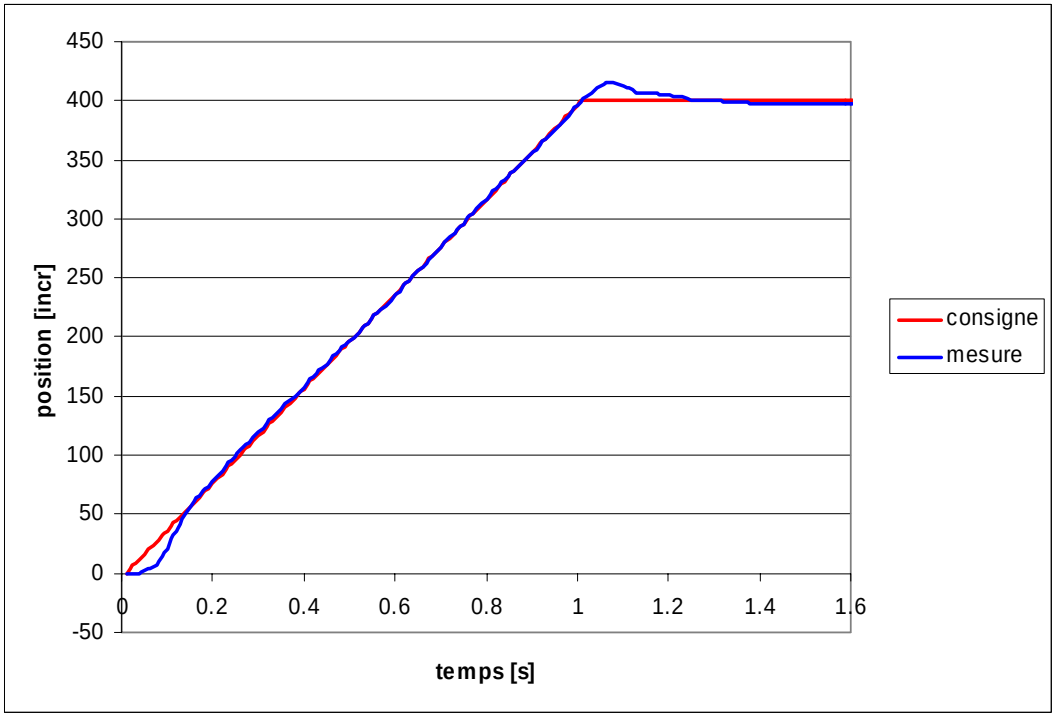


Figure 12 : rampe avec PID expérimental, pente max = 4

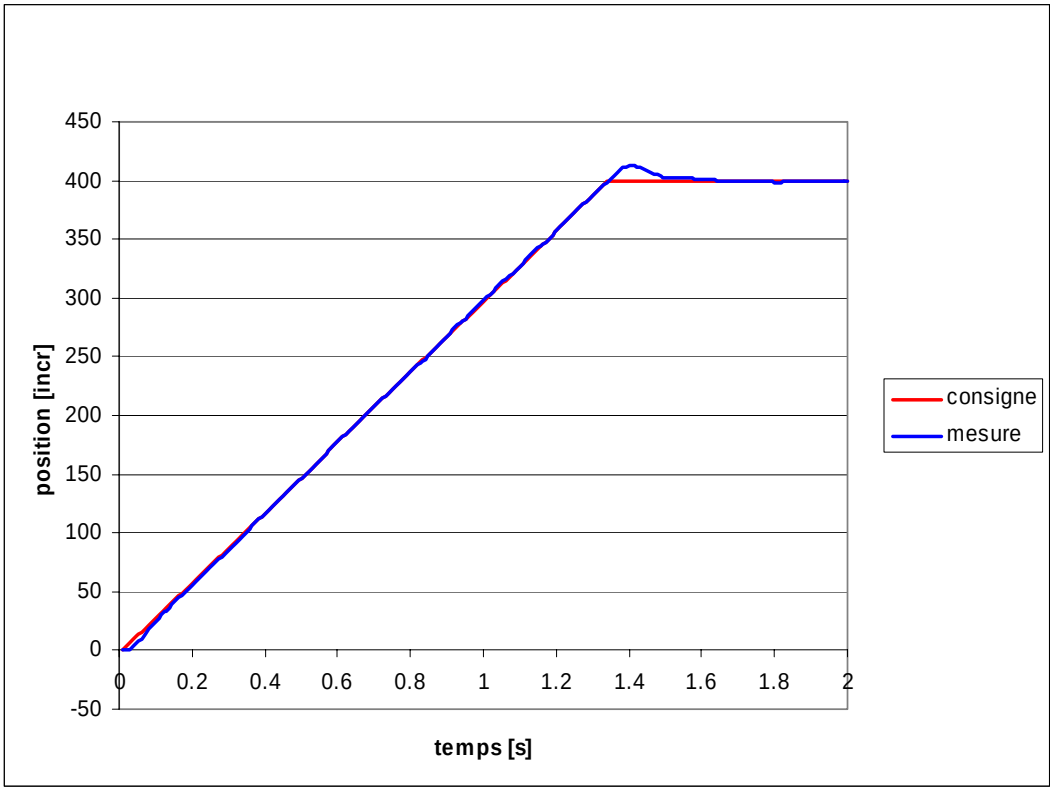


Figure 13 : rampe avec PID expérimental, pente max = 3

Commentaires

Sur cette dernière figure, on note un comportement satisfaisant. Le dépassement à la fin de la rampe est tout à fait caractéristique du comportement d'un régulateur de type PID comme celui choisi.

La dernière rampe monte jusqu'à 400 incréments, ce qui signifie un angle de balayage à vitesse constante de 72 degrés. Pour un asservissement en vitesse, c'est un ordre de grandeur raisonnable. L'angle de scanning où la vitesse doit être aussi constante que possible est représenté ci-dessous (la flèche indique la position fixe du robot pendant le match).

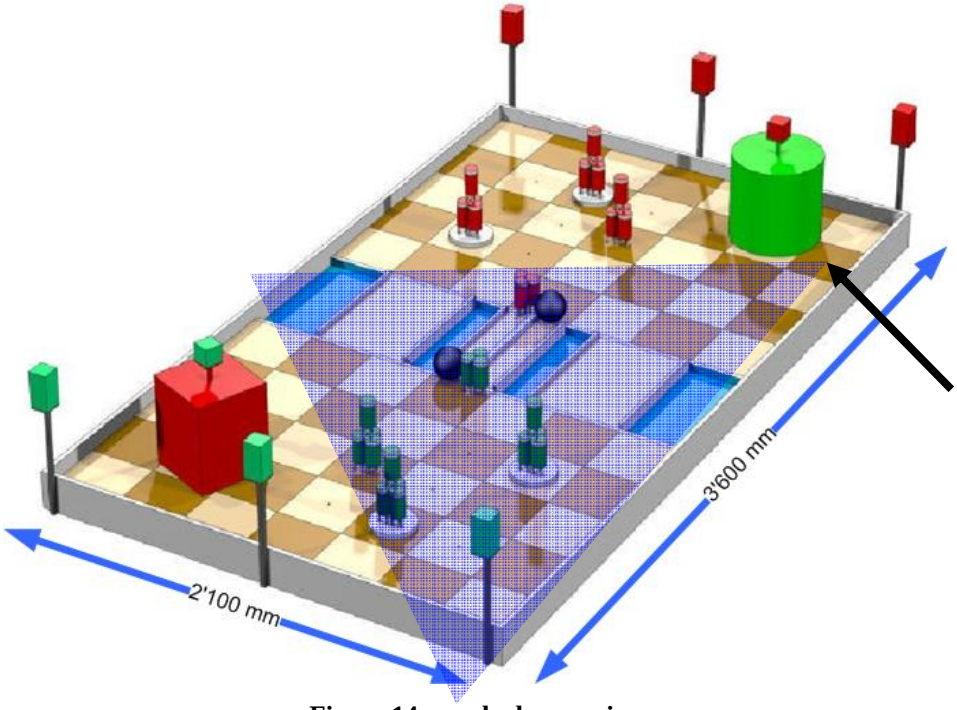


Figure 14 : angle de scanning

Notons que de faibles variations de la vitesse ne sont pas graves, puisque les valeurs mesurées avec le télémètre laser sont couplées avec celle prises par l'encodeur de la tourelle. Une vitesse variant peu est souhaitée simplement pour l'homogénéité des données prises.

Le comportement en vitesse obtenu est donc suffisant.

3.7. Estimation de la perturbation

Comme dit plus haut, l'effet du ressort gêne par son asymétrie. En observant que la contribution du ressort à la dynamique totale n'est fonction que de l'angle d'orientation de la tourelle, on peut imaginer arriver à chiffrer cette composante.

L'idée est donc ici de faire des tests sur le système réel et déterminer quelle commande est envoyée au moteur pour maintenir (régime permanent) la tourelle dans une certaine position. En reportant sur le même graphique, les valeurs maximales, minimales et moyennes d'une vingtaine de mesure, on fait ressortir clairement la caractéristique affine du ressort :

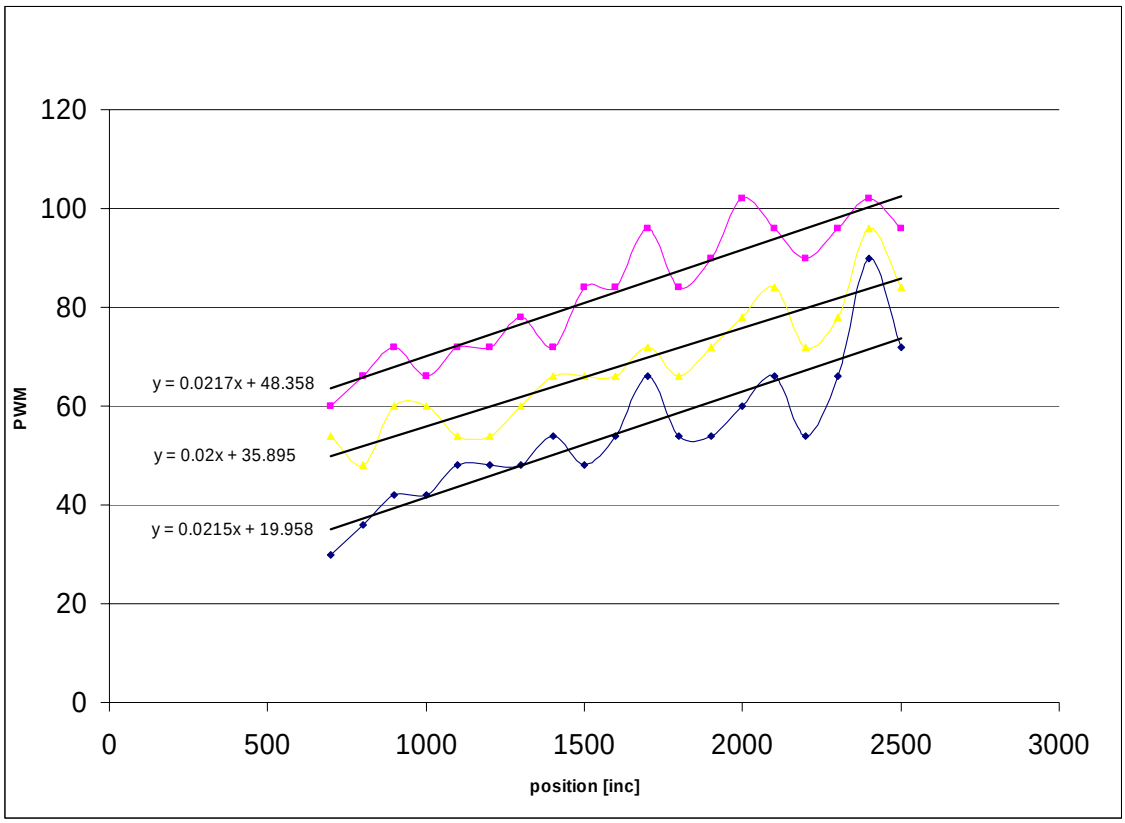


Figure 15 : commande moteur en régime permanent (système réel)

Plutôt que d'avoir une consigne *a priori* (dépendante de la consigne elle-même), on a une estimation de la perturbation due au ressort. Ceci peut facilement être représenté dans un schéma de régulation à rétroaction :

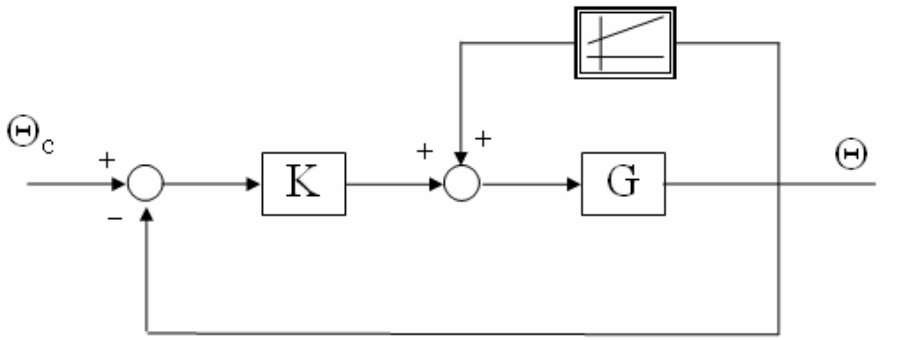


Figure 16 : schéma de régulation du système

Lorsque l'on regarde le comportement du système ainsi régulé, on remarque que le comportement est nettement meilleur à tout ce qui précède.

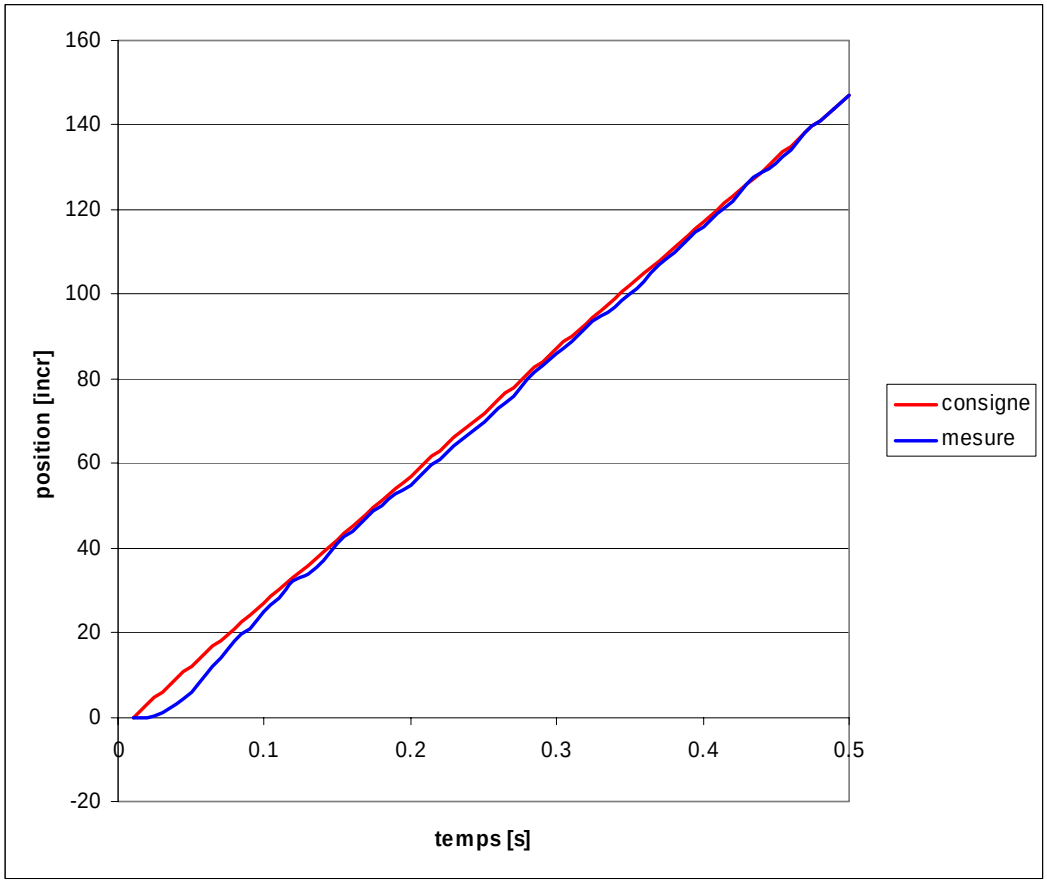


Figure 17 : réponse à une rampe, coefficients du PID expérimentaux, avec estimation de perturbation (système réel)

Un survol rapide de la figure précédent inciterait à penser que le régulateur n'est pas très bon. Il faut toutefois noter que les échelles (abscisses et ordonnées) sont clairement différentes de celles des figures précédentes.

4. Conclusions

4.1. Choix du régulateur et remarques

Le régulateur choisi est de type PID (imposé) avec estimateur de perturbation. Il satisfait le cahier des charges, à savoir un comportement correct en commande de position et de vitesse angulaire.

La partie du code en C qui concerne la régulation se trouve en annexe.

Les objectifs suivants ont également été atteints :

- l'approche expérimentale a permis d'avoir une certaine intuition de la problématique de la commande d'un système réel,
- les difficultés de la synthèse expérimentale d'un régulateur PID ont été mises en évidence,
- le contact avec d'autres personnes a permis de situer un problème d'automatique dans un contexte précis (la robotique) qui fait intervenir des connaissances différentes telles que l'informatique, l'électronique et la mécanique.

4.2. Perspectives

Comme déjà évoqué, la synthèse du régulateur PID a été faite essentiellement expérimentalement. La solution retenue est finalement assez simple mais est suffisante par rapport aux contraintes.

Même en gardant un régulateur de type PID, des améliorations seraient encore possibles, notamment par une synthèse plus propre par différentes méthodes comme par exemple dans le lieu des pôles ou dans les diagrammes de Bode. Ces concepts n'étaient pas encore connus en début de projet.

On pourrait également imaginer une approche théorique plus poussée avant de passer sur le système réel. Ceci par l'emploi de simulations numériques avec des logiciels tel que MatLab®.

Les performances du système ainsi réglé pourraient également être chiffrées par l'emploi d'une des méthodes de synthèse, notamment, des renseignements solides quant à la stabilité, précision en régime permanent, robustesse, amortissement du régime transitoire, sensibilité, bande passante.

Lausanne, le 23. juin 2005 Sylvain Rudaz

Annexe

Voici le code de régulation :

```
int regulate(void)
{
	static int consigne_position;
	int ecart_position = 0;
	int Kp = 6;
	int milfoishsurTi = 60;
	int Tdsurh = 8;
	int a_priori = 0;  // il s'agit en fait de l'estimateur de perturbation !

	static long int ui;
	static long int old_ui;
	static long int ud;
	static long int old_ud;
	static int old_ecart_position;
	static int compteur_variation;

	static int compteur_tableau;
	const int MAX_PWM_TURRET = 255;
	const int MAX_1000fois_ui = 30000;
	const int TURRET_MAX_DELTA_SPEED = 3;

	/*
	 * mot0_turret regulation
	 */

	//envoi de l'ecart à la position de consigne
	if (abs(mot0_turret.wanted_pos - mot0_turret.increments) < 20) {
		mot0_turret.variation = mot0_turret.wanted_pos - mot0_turret.increments;
	} else {
		mot0_turret.variation = 100;
	}

	// recalcul de la vitesse de la tourelle
	if (mot0_turret.wanted_speed == 0) {
		mot0_turret.wanted_speed = TURRET_MAX_DELTA_SPEED;
	} else if (abs(mot0_turret.wanted_speed) > TURRET_MAX_DELTA_SPEED) {
		mot0_turret.wanted_speed = TURRET_MAX_DELTA_SPEED;
	}
	mot0_turret.wanted_speed = abs(mot0_turret.wanted_speed); //mot speed positif

	// création des rampes de consignes, positive et négative
	if (mot0_turret.wanted_pos > mot0_turret.increments) {
		// création de la rampe de consigne, sens positif
		if (compteur_consigne < mot0_turret.wanted_pos) {
			compteur_consigne += mot0_turret.wanted_speed;
		} else {
			compteur_consigne = mot0_turret.wanted_pos;
		}
	} else {
		// création de la rampe de consigne, sens négatif
		if (compteur_consigne > mot0_turret.wanted_pos) {
			compteur_consigne -= mot0_turret.wanted_speed;
		} else {
			compteur_consigne = mot0_turret.wanted_pos;
		}
	}

	// mise à jour de la consigne
	consigne_position = compteur_consigne;
	int_affichage = consigne_position;


	//estimation de perturbation
	a_priori = mot0_turret.increments / 50 + 50;
```

```
	//PID
	ecart_position = consigne_position - mot0_turret.increments;
	ui = old_ui + ecart_position * milfoishsurTi;
	ud = Tdsurh * (ecart_position - old_ecart_position);

	//ARW ui
	if (ui > MAX_1000fois_ui)
		ui = MAX_1000fois_ui;
	else if (ui < -MAX_1000fois_ui)
		ui = -MAX_1000fois_ui;

	mot0_turret.wanted_speed = (ecart_position + ui / 1000 + ud) * Kp + a_priori;

	//mises a jour
	old_ui = ui;
	old_ud = ud;
	old_ecart_position = ecart_position;


	if (mot0_turret.wanted_speed >= 0) {
		// we put the mod for forward both 01
		motors_1->control &= ~MOT_LEFT_MOD_1;
		motors_1->control |= MOT_LEFT_MOD_0;

		if (mot0_turret.wanted_speed > MAX_PWM_TURRET)
			mot0_turret.wanted_speed = MAX_PWM_TURRET;
	} else {
		// we put the mod for backwards both 10
		motors_1->control &= ~MOT_LEFT_MOD_0;
		motors_1->control |= MOT_LEFT_MOD_1;

		if (mot0_turret.wanted_speed < -MAX_PWM_TURRET)
			mot0_turret.wanted_speed = -MAX_PWM_TURRET;
	}
	// directly sets the PWM
	motors_1->mot_pwm_l_icnt = abs(mot0_turret.wanted_speed);


	if (arret_moteurs) {
		motors_0->mot_pwm_l_icnt = 0;
		motors_0->mot_pwm_r_icnt = 0;
		motors_1->mot_pwm_l_icnt = 0;
		motors_1->mot_pwm_r_icnt = 0;
		printf("interrupteur d'urgence acitonne\r\n");
	}
}

int encoder_update(void)
{
	unsigned long temp_enc;

	/*
	 * update for the turret motor
	 */
	mot0_turret.last_increments = mot0_turret.increments;

	mot0_turret.real_speed = mot0_turret.increments - mot0_turret.last_increments;
	temp_enc = ((unsigned long) motors_0->encl_high) << 16;
	temp_enc += (unsigned int) motors_0->encl_low;
	mot0_turret.increments = (long) temp_enc - mot0_turret_pos_init;
}
```

Bibliographie

[1] R. LONGCHAMP, *Commande numérique de systèmes dynamiques*, Presses polytechniques et universitaires romandes, Lausanne, 1995.