Contribution à l'interaction commande/ordonnancement

David Robert

▶ **To cite this version:**

David Robert. Contribution à l'interaction commande/ordonnancement. Automatique / Robotique. Institut National Polytechnique de Grenoble - INPG, 2007. Français. NNT : . tel-00207709

HAL Id: tel-00207709
https://theses.hal.science/tel-00207709

Submitted on 18 Jan 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Institut National Polytechnique de Grenoble

No. attribué par la bibliothèque
|_|_|_|_|_|_|_|_|_|_|

THESE

pour obtenir le grade de

DOCTEUR DE L'INPG

***Spécialité :* AUTOMATIQUE-PRODUCTIQUE**

préparée au Laboratoire d'Automatique de Grenoble

dans le cadre de l'École Doctorale :
Électronique, Électrotechnique, Automatique, Traitement du Signal

présentée et soutenue publiquement

par

David ROBERT

le 11 Janvier 2007

Titre :

Contribution à l'interaction commande/ordonnancement

Directeurs de thèse :

M. Olivier SENAME	(LAG - INPG)
M. Daniel SIMON	(INRIA Rhône-Alpes)

JURY :

M. Carlos CANUDAS de WIT	Président
M. Philippe CHEVREL	Rapporteur
M. Albert BENVENISTE	Rapporteur
M. Pedro ALBERTOS	Examinateur
M. Olivier SENAME	Directeur de thèse
M. Daniel SIMON	Co-directeur de thèse

Table des matières

Remerciements

Durant ces trois dernières années j'ai eu l'occasion de rencontrer de nombreuses personnes et je m'excuse d'avance pour toutes celles que je vais inévitablement oublier.

Je tiens tout d'abord à remercier Olivier SENAME et Daniel SIMON qui m'ont encadré durant ces trois dernières années. Leur calme et leur bonne humeur ont installé un climat de travail particulièrement agréable. Ils ont toujours été prêts à répondre à mes questions et m'ont laissé une très grande liberté dans mon travail.

Je tiens aussi à remercier Albert BENVENISTE pour son regard très objectif sur mon travail en tant que rapporteur. Ces critiques ont été très constructives et la qualité du mémoire n'a pu que progresser. Merci aussi à Philippe CHEVREL d'avoir consacré du temps à relire mon travail. Son expertise a apporté des idées très intéressantes pour la poursuite du sujet.

Merci à Carlos CANUDAS de WIT qui me fait le plus grand plaisir d'être le président de mon jury. Merci aussi à Pedro ALBERTOS qui se déplace d'Espagne pour découvrir mon travail.

Je voudrais aussi remercier tous les membres du laboratoire d'automatique de Grenoble, les chercheurs, ingénieurs, membres de l'administration, pour leurs commentaires, leur aide et sans qui une thèse ne serait pas possible. Pour les mêmes raisons, je remercie aussi les membres de l'équipe POP ART de l'INRIA Rhône-alpes.

Je tiens à remercier mes différents voisins de bureau et notamment Ahmed, Alessandro, Sameh, Trinh, Sylvie, Gwénaël, Hamoudi, Emil, ... Ils ont connu mes coups de blues, mes "pétages de plomb", mais nous avons surtout passé de très bons moments ensemble. Un très grand merci à tous mes amis thésards, stagiaires ainsi que leur moitié, et notamment, Sophie, Matthieu, Jean-Matthieu, Rodolphe, Christophe, Marc, Denis, Jonathan, Pietro tous les bipopistes, tous les popartistes ; ce sont trois années très riches en rencontres !

Enfin un grand merci à ma famille sans qui je n'aurais pas pris la décision, il y a trois ans, de tenter l'aventure du doctorat. Un grand merci à Blandine qui m'a soutenu durant les moments difficiles et pour qui j'ai tenu le coup.

Merci à tous ...

Chapitre 1

Introduction

Les systèmes embarqués sont désormais omniprésents dans notre vie quotidienne. De nombreux appareils qui nous entourent contiennent un ou plusieurs microprocesseurs. Alors qu'ils sont indispensables aux derniers produits de haute technologie, comme les téléphones portables ou les écrans plats, on en trouve maintenant dans des appareils plus traditionnels tels que les machines à laver.

La classe des systèmes embarquées se distingue des systèmes électroniques ou informatiques plus traditionnels par des contraintes sur la place, le poids, la source d'énergie, pour n'en citer que quelques unes. Les limitations techniques qui en découlent portent notamment sur la puissance de calcul, l'espace mémoire de stockage ou la bande passante d'un canal de communication.

L'apparition des systèmes embarqués dans les produits de grande consommation ajoute une nouvelle contrainte : le coût. Alors que le prix du calculateur d'un avion civil, du TGV ou d'une centrale nucléaire est justifié par des arguments de sécurité, de fiabilité et de longévité, celui d'un appareil photo, d'une carte à puce ou d'une voiture doit être en équation avec le coût du produit. Bien que les processeurs progressent, il n'est pas économiquement viable de prendre un calculateur surpuissant ou même d'en utiliser plusieurs, là où une conception soignée se suffirait d'un seul de puissance moyenne. Le coût devient alors un argument dans l'optimisation de l'usage des ressources.

Parmi l'ensemble des systèmes embarqués, nous nous intéressons ici aux applications de contrôle/commande. Cette classe de systèmes se distingue par la présence de contraintes temporelles fortes. En effet ils doivent réagir à des événements extérieurs dans des temps pré-définis. Ils sont présents dans de nombreuses applications, la voiture en est un très bon exemple avec plusieurs dizaines de calculateurs interconnectés, pour des fonctions comme le contrôle du moteur, de la trajectoire, de l'ABS, la régulation de la vitesse ou de la climatisation.

La conception traditionnelle des systèmes de contrôle/commande, présents notamment

dans l'industrie, la production d'énergie, le transport ferroviaire, supposait des ressources de calcul et de communication quasi illimitées. Les systèmes embarqués bousculent ces habitudes et il devient nécessaire de prendre en compte les contraintes et l'optimisation de l'usage des ressources dès la conception du système.

Dans ce chapitre, après des rappels sur la commande d'un système par ordinateur, nous présentons les limites de la conception actuelle et les interactions entre l'ordonnancement d'un système informatique et le contrôle d'un procédé physique. Enfin nous introduirons les différents chapitres de ce document.

1.1 Contrôle par ordinateur d'un procédé dynamique

1.1.1 Conception traditionnelle d'une commande par ordinateur

La structure classique d'une commande digitale est représentée sur la figure 1.1. Le procédé dynamique à contrôler est muni de capteurs et d'actionneurs qui sont connectés à l'unité de calcul. L'exécution du contrôleur est composée de 3 phases : l'acquisition des mesures, le calcul de la nouvelle commande et la mise à jour de l'actionneur.

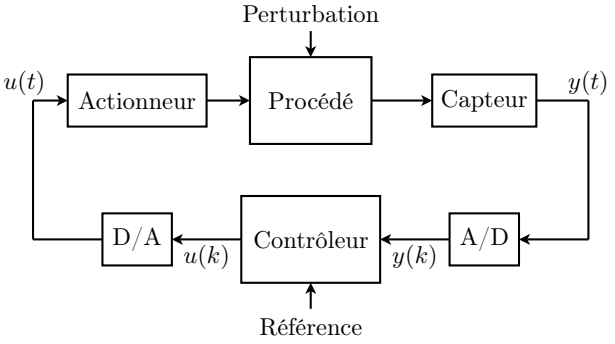


FIG. 1.1 – Schéma d'une commande digitale

Ces trois opérations sont généralement exécutées en séquence. Ainsi l'acquisition des mesures, au travers de convertisseurs analogiques/numériques, est déclenchée par une horloge périodique. Dès la disponibilité de ces mesures, la nouvelle commande est calculée. Puis une conversion numérique/analogique actualise la commande, qui est maintenue constante par un bloqueur d'ordre zéro jusqu'à la prochaine mise à jour.

Idéalement ces trois opérations sont parfaitement périodiques et synchronisées conduisant à une période constante et une latence entre la mesure et la mise à jour de la commande nulle ou constante.

Dans ce contexte, la loi de commande, c'est à dire l'algorithme qui calcule la nouvelle commande en fonction des mesures, est conçue par l'usage d'une des nombreuses méthodes

de synthèse de contrôleurs (placement de pôles, commande optimale, commande robuste, ...). Le correcteur peut être obtenu directement en temps discret ou via la discrétisation d'un correcteur à temps continu. Dans les deux cas les approches traditionnelles supposent l'échantillonnage parfaitement périodique et la latence entrée/sortie nulle ou constante.

La période d'échantillonnage est choisie en fonction de la dynamique du procédé et des objectifs visés en boucle fermée. Pour cela il existe des critères empiriques et notamment celui d' Åström and Wittenmark (1997) :

$$0.2 \leq w_0 h \leq 0.6 \tag{1.1}$$

où h est la période d'échantillonnage et w_0 la bande passante de la boucle fermée. Ce critère, plus conservatif que le critère de Shannon, garantit une marge de phase minimale.

La loi de commande est implémentée soit sur un gros calculateur dont les ressources peuvent être partagées avec d'autres activités, soit sur un micro calculateur dédié à cette fonction. Dans le premier cas l'usage d'un système d'exploitation temps réel multi-tâches se généralise, facilitant l'écriture de programmes concurrents. Dans le second cas le développement de solutions sur mesure est souvent suffisant.

1.1.2 Ordonnancement temps réel

La théorie de l'ordonnancement temps-réel a pour objectif de trouver l'ordre d'exécution d'un ensemble de tâches qui garantit le respect de leurs contraintes temporelles. Les politiques d'ordonnancement sont regroupées en deux familles : les ordonnancements statiques et dynamiques.

L'ordonnancement statique est une approche hors ligne qui utilise des algorithmes d'optimisation pour générer un scénario cyclique. L'instant d'activation de chaque tâche est défini très précisément. A l'exécution, le scénario est scrupuleusement répété sans adaptation possible. Les avantages sont d'une part la facilité d'analyse du respect des contraintes temporelles et d'autre part une implémentation simplifiée. L'un des inconvénients est que l'ordonnancement peut être long et difficile à obtenir. Il nécessite la connaissance très précise de chaque tâche et notamment sa durée d'exécution maximale. La longueur du scénario peut être élevée si les tâches ont des périodes ou des exécutions très disparates. Enfin il est difficile de prendre en compte des tâches d'exécution non périodique.

L'ordonnancement dynamique est une approche en ligne. Le séquencement des tâches n'est pas défini à l'avance et lors de l'exécution, la tâche à activer est choisie selon une politique d'ordonnancement. Cette approche rend l'analyse plus complexe mais apporte plus de flexibilité. Les politiques les plus classiques ont été introduites par Liu and Layland (1973) : priorité fixe (fixed priority ou FP) et échéance la plus proche (earliest-deadline-first ou EDF). Ces approches reposent sur le modèle d'une tâche τ_i décrite par :

- une période T_i,
- une échéance relative D_i,
- un temps d'exécution maximal C_i.

La figure 1.2 illustre les attributs temporels d'une tâche périodique. A ce modèle simplifié, il faut ajouter l'hypothèse de tâches indépendantes, c'est à dire qu'elles ne communiquent pas et qu'elles ne partagent pas de ressources autres que le processeur.

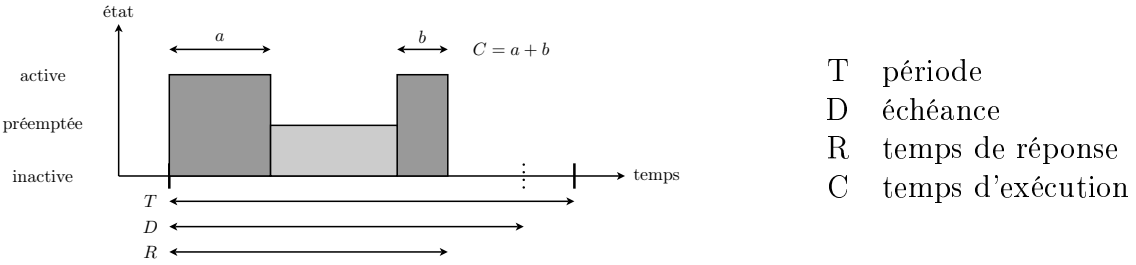


FIG. 1.2 – Attributs temporels d'une tâche

Ordonnancement à priorité fixe

Cette politique d'ordonnancement est la plus simple et la plus répandue des ordonnancements dynamiques. Chaque tâche se voit attribuée une priorité à la conception du système. A l'exécution, l'unité de calcul est affectée à la tâche prête de plus haute priorité. Si une tâche de plus faible priorité est en cours d'exécution au moment de cette décision, elle est alors préemptée par la plus prioritaire, c'est à dire qu'elle est interrompue jusqu'à redevenir la tâche prête la plus prioritaire.

Il existe plusieurs critères pour choisir la priorité des tâches :

- l'urgence relative : ce choix est arbitraire et dépend de considérations d'urgence entre les tâches, propres à une application donnée.
- rate-monotonic (RM) : la priorité est inversement proportionnelle à la période T_i. Ainsi plus la période est petite et plus la priorité est élevée.
- deadline-monotonic (DM) : la priorité est inversement proportionnelle à l'échéance relative D_i.

Liu and Layland (1973) ont montré, sous les hypothèses du modèle simplifié présenté plus haut, que RM est une police d'ordonnancement optimale, c'est à dire qu'un ensemble de tâches qui n'est pas ordonnançable selon cette politique ne pourra pas l'être sous tout autre politique à priorité fixe. Leung and Whitehead (1982) ont montré que sous les mêmes hypothèses, DM est aussi optimale. Notons que RM est un cas particulier de DM avec l'échéance égale à la période.

Connaissant les attributs de chaque tâche (période, échéance, temps d'exécution dans le pire cas et priorité) il est possible d'analyser le respect des contraintes temporelles. Liu and

Layland (1973) montrent, dans le cas de rate-monotonic, qu'une condition suffisante mais pas nécessaire pour que les échéances d'un système composé de n tâches soient respectées est :

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n(2^{1/n} - 1) \tag{1.2}$$

où U est la charge processeur comprise entre 0 et 1. La limite de l'expression de droite tend vers $\ln 2 \approx 0.69$ quand n tend vers l'infini.

Un test plus approfondi peut être obtenu en calculant le temps de réponse maximal R_i de chaque tâche et en vérifiant si $R_i \leq D_i, \forall i$. Sous les mêmes hypothèses que précédemment, le temps de réponse est maximal quand toutes les tâches démarrent au même moment. Dans ce cas le temps de réponse maximal de la tâche τ_i est défini par l'expression récursive suivante (Joseph and Pandya, 1986) :

$$R_i = C_i + \sum_{j \in h_p(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j \tag{1.3}$$

où $h_p(i)$ est l'ensemble des tâches de priorité supérieure à celle de τ_i et $\lceil x \rceil$ est la fonction arrondi à l'entier supérieur.

Ordonnancement EDF

Sous la politique EDF, la tâche active possède l'échéance absolue la plus proche. Contrairement à l'ordonnancement FP, il n'y a pas de priorités définies à l'avance.

Si les tâches sont indépendantes et que $D_i = T_i$, une condition nécessaire et suffisante pour que toutes les tâches respectent leur échéance est la suivante (Liu and Layland, 1973) :

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1 \tag{1.4}$$

Dans ce cas, EDF est plus efficace que rate-monotonic puisque l'utilisation du processeur est plus élevée. Le cas plus complexe où $D_i \leq T_i$ est traité dans (Stankovic et al., 1998). Néanmoins EDF a certains inconvénients (Burns and Wellings, 2001) et notamment une implémentation plus complexe, un écroulement du système en cas de surcharge et une estimation de l'échéance absolue parfois difficile à obtenir en ligne.

Le calcul du temps de réponse pour EDF est plus complexe que pour FP et a été établi par Stankovic et al. (1998).

Extensions

Nous venons d'introduire succinctement les politiques FP et EDF sous des hypothèses restrictives. L'analyse de la politique FP a été améliorée pour prendre en compte notamment les interactions entre tâches et les tâches apériodiques. L'ordonnancement EDF a été étendu dans des directions similaires. Pour plus de détails, le lecteur peut se reporter à l'article de Sha et al. (2004).

Nous retiendrons de cette section que dans les conditions restrictives du modèle de tâche présenté plus haut, il est possible de garantir le respect des échéances par un simple test sur le taux d'utilisation du processeur. Néanmoins toutes ces études a priori nécessitent la connaissance du temps d'exécution dans le pire cas (WCET), qui en pratique est difficile à obtenir avec précision. Enfin notons que les modèles de tâches prennent rarement en compte l'indéterminisme apporté par l'architecture matériel et par l'exécutif temps réel.

1.2 Conception conjointe commande/ordonnancement

Lors du développement d'une application de contrôle/commande, il y a traditionnellement une séparation forte entre la conception de la loi de commande et son implémentation sur le calculateur.

D'un côté, une application de contrôle/commande est souvent prise comme exemple d'un système temps réel dur par la communauté informaticienne. Une loi de commande est alors assimilée à une tâche périodique, de période constante et de priorité fixe. Son temps d'exécution est borné et son échéance, inférieure à la période, doit être strictement respectée. Ces hypothèses sont suffisantes pour concevoir l'implémentation et étudier l'ordonnancement des tâches afin de garantir le respect des échéances.

D'un autre côté la communauté automaticienne suppose l'échantillonnage équidistant, et la latence entrée-sortie nulle ou constante. Ces hypothèses conviennent parfaitement aux outils d'analyse et de synthèse dédiés aux contrôles des systèmes échantillonnés.

Ces hypothèses ont permis aux deux communautés de se concentrer respectivement sur leur domaine de recherche. D'un côté le développement de modèles de tâches et de politiques d'ordonnancement qui garantissent le respect des échéances et de l'autre celui de méthodes pour concevoir les régulateurs à temps discret.

L'usage montre que cette séparation est tout à fait viable dans bon nombre de situations. Néanmoins l'exigence toujours grandissante en terme de qualité de commande (vitesse, précision, robustesse, ...) ou d'implémentation (contraintes des systèmes embarqués, coûts, ...) de même que l'augmentation de la complexité des systèmes et l'usage des réseaux de communication font que cette méthodologie atteint ses limites.

Ainsi, assimiler une loi de commande à un système temps réel dur est souvent trop

restrictif. En effet pour ce type de système le dépassement temporaire d'une échéance ne conduit pas obligatoirement à la catastrophe. Grâce à la robustesse apportée par la contre réaction, une échéance ratée, qui se traduit par un retard temporaire ou la perte d'une commande, va dégrader la performance mais le système reste stable. Ainsi, autoriser un taux de dépassement d'échéances permettrait d'assouplir les contraintes d'ordonnancement pour mieux utiliser les ressources. L'hypothèse même d'une tâche périodique ne convient pas toujours, on peut citer l'exemple des moteurs automobiles où certaines lois de commande sont synchronisées sur la vitesse de rotation du moteur.

D'autre part considérer l'échantillonnage parfaitement périodique ou négliger le temps de calcul est une vision simplifiée de la réalité. Le partage du processeur entre plusieurs activités introduit des variations dans la période d'échantillonnage, de même que le calcul de la commande ne peut être de durée nulle. Or un retard entre la mesure et la mise à jour de la commande, ou une gigue sur l'instant d'échantillonnage dégradent la qualité de la commande. La prise en compte de ces perturbations peut améliorer la commande et dans des cas difficiles peut s'avérer incontournable.

Enfin un des inconvénients est que cette séparation conduit à une approche conservative quant à l'usage des ressources. Ainsi pour un calculateur donné, le respect strict des échéances conduit à limiter la période ou le temps d'exécution de l'algorithme et donc indirectement sa performance. Si l'on souhaite une meilleure commande, les solutions sont alors de prendre un calculateur plus performant ou d'optimiser conjointement la commande et l'ordonnancement. Les contraintes de "l'embarqué" ont fait naître un intérêt pour la seconde approche.

1.2.1 Interaction commande/ordonnancement

La figure 1.3 illustre l'exécution d'une tâche de commande sur un processeur partagé entre plusieurs activités, via l'usage d'un ordonnancement préemptif. On suppose qu'à chaque exécution du contrôleur, ce dernier fait l'acquisition des mesures puis calcule la nouvelle commande et enfin met à jour les actionneurs.

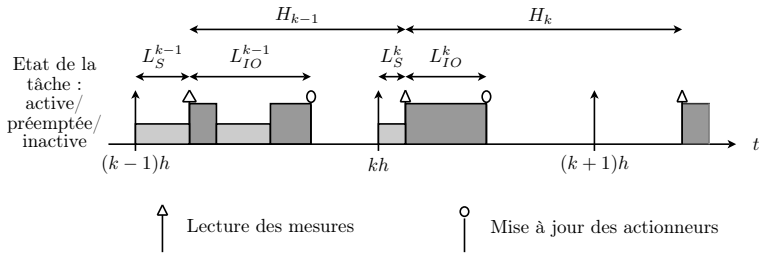


FIG. 1.3 – Exemple d'exécution d'une tâche de commande

Cet exemple fait apparaître trois paramètres temporels qui établissent un lien entre l'ordonnancement et la loi de commande.

L_s **latence d'échantillonnage** C'est la durée qui sépare l'instant théorique (kh) où les mesures devraient être échantillonnées et l'instant réel de l'acquisition. Ce délai est la conséquence d'une préemption qui a débuté avant l'activation de la tâche.

L_{io} **latence entrée-sortie** C'est le délai entre les mesures et la mise à jour des actionneurs. Il est composé du temps d'exécution de l'algorithme de commande et de la durée des préemptions qui apparaissent après le début de l'exécution de la tâche.

H **période effective** C'est la durée effective entre deux échantillonnages consécutifs des mesures.

Si les capteurs et les actionneurs sont déportés et connectés au calculateur par un réseau de communication, les temps de transmissions vont intervenir dans les valeurs de L_S et L_{IO}. Dans le cas d'une étude plus fine, on peut ajouter les durées des conversions analogique/numérique et des traitements de l'exécutif temps réel.

Les effets de ces perturbations temporelles sur la qualité d'une commande sont difficiles à déterminer. D'une part il est complexe d'analyser l'influence de la politique d'ordonnancement sur leurs valeurs, d'autre part l'action des ces trois perturbations sur le comportement d'une commande est difficile à quantifier.

Latence entrée/sortie : L_{IO}

Idéalement la latence entrée/sortie devrait être constante. Néanmoins, les mémoires caches de certains calculateurs et les lois de commande utilisant une optimisation en ligne ou des changements de mode produisent des variations du temps d'exécution de l'algorithme. De plus la variation du temps d'exécution des tâches plus prioritaires modifie la durée des préemptions. Il en résulte que cette latence est composée d'une valeur moyenne à laquelle s'ajoute une déviation communément appelée gigue d'entrée/sortie.

L'effet sur la commande est assimilable à un retard en entrée du système. Ce retard induit une perte de phase et donc une dégradation de la performance en boucle fermée (Åström and Wittenmark, 1997).

Période effective H et latence d'acquisition L_S

La période effective et la latence d'acquisition sont étroitement liées. En effet si la latence d'acquisition est constante, la période effective reste identique à la période désirée. A l'inverse une variation de L_S produit une gigue sur l'instant d'acquisition et donc une gigue sur la période effective appelée gigue d'échantillonnage.

L'effet de la gigue d'échantillonnage sur la commande est difficile à quantifier car il est difficile à modéliser. En pratique, une variation de la période d'échantillonnage inférieure à 10 % semble négligeable. Néanmoins des valeurs plus importantes peuvent la dégrader comme l'illustre Cervin (2003).

1.2.2 Ordonnancement pour la commande

Le partage d'une unité de calcul entre plusieurs activités est à l'origine des perturbations temporelles énoncées précédemment, il est logique de tenter de résoudre le problème à la source et donc d'agir sur l'ordonnancement. Notons que si l'on utilise un calculateur par loi de commande, comme il est tout à fait envisageable en utilisant des micro-contrôleurs, le problème ne se pose pas.

En première approche, on peut affecter la plus haute priorité à l'algorithme de commande, ce qui n'est valable que dans le cas d'une seule commande. On peut aussi utiliser un ordonnancement non préemptif mais dans ce cas, l'analyse du respect des échéances d'un ensemble de tâches est plus difficile.

Une autre approche consiste à décomposer l'algorithme de commande en plusieurs parties ou sous-tâches. L'acquisition peut être réalisée par une tâche dédiée de très haute priorité et la mise à jour de la commande retardée jusqu'à la fin de la période. Il en résulte une latence d'une période et une gigue quasi nulle. L'intérêt de cette approche dépend alors de la capacité à compenser la forte valeur de latence. Cervin (1999) propose une décomposition en deux parties : le calcul de la nouvelle commande et la mise à jour de l'état interne du correcteur. En leur attribuant une priorité ou une échéance différente, suivant l'usage de la politique EDF ou FP, cette approche est plus performante qu'un contrôleur monotâche ou qu'un contrôleur dont la mise à jour de la commande est retardée à la prochaine mesure. Dans (Balbastre et al., 2004) les tâches de commandes sont décomposées en trois sous-tâches regroupées par groupe de priorités différentes : mesure (priorité moyenne), calcul de la commande (priorité basse) et mise à jour des sorties (priorité haute). Leurs résultats, qui prend en compte le partage des ressources entre plusieurs lois de commande, montrent une réduction de l'amplitude de la gigue et une augmentation de la latence moyenne.

Le concept d'ordonnancement à réservation de ressources, et notamment le serveur à bande passante constante (constant bandwidth server ou CBS) introduit par Abeni and Buttazzo (1998) permet de diviser un processeur en plusieurs processeurs virtuels. Chaque processeur virtuel se voit attribuer une quantité limitée de ressources qu'il ne peut pas dépasser. Toute surcharge est ainsi confinée à un processeur virtuel sans affecter les autres. Cette approche a été étendue aux tâches de commande par Cervin and Eker (2005) sous le nom de serveur de commandes. Ces dernières sont décomposées en plusieurs segments. Les échanges d'entrées/sorties s'effectuent à des instants prédéfinis ce qui rend la latence entrée-sortie constante. L'approche permet d'établir un lien simple entre la performance de la commande et le facteur d'utilisation du serveur de commandes.

L'usage d'un ordonnancement adaptatif ou régulé permet de compenser dynamiquement les dégradations de la commande. L'idée est d'adapter en ligne l'ordonnancement de plusieurs tâches de commande à partir des mesures des perturbations temporelles et de la performance des commandes. L'objectif est d'optimiser la performance des contrôleurs sous contrainte d'exécution. Cette approche nécessite un executif temps réel, permettant les me-

sures des perturbations temporelles, et la modélisation de leurs effets sur la commande. Nous reparlerons de cette approche dans le chapitre suivant.

Enfin notons qu'il existe des approches totalement différentes de l'implémentation de systèmes de contrôle/commande. Citons notamment les langages synchrones tels que Lustre, Esterel et Signal ou les architectures TTA (Time triggered architecture).

1.2.3 Commande sensible aux effets de l'implémentation

Bien que les perturbations temporelles soient issues de l'ordonnancement et qu'il est logique d'intervenir sur ce dernier pour les réduire, étudier ce problème du point de vue de l'automatique a plusieurs intérêts. D'une part, ces perturbations sont diminuées par un ordonnancement adapté sans être complètement éliminées. D'autre part, l'usage des réseaux de communication produit des perturbations similaires indépendantes de l'ordonnancement. Enfin il est intéressant d'étudier des commandes dont l'échantillonnage n'est pas totalement périodique, ceci pour apporter plus de flexibilité.

Du point de vue de la commande, la latence entrée/sortie est assimilable à un retard en entrée du procédé. Ce retard induit une perte de phase et donc une dégradation de la performance en boucle fermée. La théorie de la commande s'affranchit aisément d'un retard constant et connu, par l'augmentation du vecteur d'état (Åström and Wittenmark, 1997). Le contrôle d'un système comportant un retard inconnu, borné ou non est plus délicat (Richard, 2003). Le résultat d'analyse de stabilité en présence de retards variables de Kao and Lincoln (2004) a été repris par Cervin et al. (2004) pour définir une marge de gigue, qui est une extension de la marge de retard au cas des retards à temps-variant. La boîte à outils Jitterbug (Lincoln and Cervin, 2002), développée pour le logiciel Matlab, offre une analyse stochastique de la performance d'une commande sous diverses conditions temporelles et notamment la variation de la latence entrée/sortie. Dans (Sename et al., 2003) nous utilisons un retour d'état stabilisant, valable pour un retard variable mais borné, afin de garantir la stabilité d'un procédé commandé par réseau. Dans ce cas les retards proviennent du réseau ou de la latence entrée/sortie de l'algorithme de commande.

Les perturbations temporelles peuvent aussi être considérées comme des perturbations paramétriques ou des dynamiques négligées. Cela permet d'utiliser les outils de la commande robuste tels que la commande H_∞ et la μ-analyse.

Si les perturbations temporelles sont difficilement quantifiables à la conception, il est possible de les compenser à l'exécution de la commande. Nilsson (1998), dans un contexte de commande par réseau, présente un contrôleur dépendant de la latence entre le capteur et le contrôleur. Cette approche n'est pas directement applicable au problème de la latence entrée-sortie faute d'en connaître la valeur à l'avance. Lincoln (2002) propose un compensateur dont la structure est proche d'un prédicteur de Smith qui utilise la valeur précédente de la latence.

Une technique simple pour compenser la gigue d'acquisition est d'adapter le correcteur

à la valeur courante de la période. Relativement facile à mettre en oeuvre dans le cas d'un PID, cette approche est plus difficile avec des correcteurs d'ordre plus élevé ou issus d'une synthèse à temps discret. Des solutions sur mesure ont été proposées par Albertos and Crespo (1999); Martí et al. (2001). Une solution reposant sur un filtre de Kalman à temps variant est présenté par Nilsson (1998), mais l'implémentation nécessite des calculs coûteux. Dans les chapitres suivants, nous reviendrons sur la synthèse d'un correcteur dynamique à période variable.

1.2.4 Etude conjointe commande/ordonnancement

Nous venons de décrire des solutions qui portent individuellement sur l'ordonnancement ou sur la commande. Le rapprochement entre les communautés des systèmes temps réel et de l'automatique a donné lieu à des approches mixtes où l'objectif est d'étudier conjointement la commande et l'ordonnancement lors de la conception d'un contrôle par ordinateur.

L'un des premiers travaux est celui de Seto et al. (1996). Il s'agit de l'optimisation hors ligne des périodes de plusieurs lois de commande sous contrainte d'ordonnançabilité. Chaque loi de commande est munie d'une fonction de coût, approximée par une exponentielle de la période. L'objectif est de minimiser la somme des fonctions de coût en garantissant que le taux d'utilisation du processeur est inférieur à une borne. La méthode est appliquée aux politiques FP/RM et EDF. Bini and Natale (2005) reprennent la même approche mais affinent la contrainte d'ordonnançabilité, dans le cas d'un ordonnancement FP, pour permettre l'ordonnancement d'un plus grand nombre de tâches. Néanmoins la résolution du problème d'optimisation devient plus complexe.

Dans (Ryu et al., 1997; Ryu and Hong, 1998), la latence entrée/sortie et la période de plusieurs lois de commande sont optimisées conjointement. Pour cela les auteurs expriment la performance des commandes (erreur statique, dépassement, temps de montée et de réponse) en fonction de la période et de la latence. Une heuristique d'optimisation maximise la performance des commandes sous contrainte d'ordonnançabilité.

La recherche d'une séquence statique d'ordonnancement de plusieurs commandes LQ a été présentée par Lincoln and Bernhardsson (2000). La solution, qui repose sur la résolution d'un problème d'optimisation combinatoire, produit une séquence d'activation des contrôleurs et non une période à leur attribuer. Dans (Rehbinder and Sanfridson, 2000), une approche similaire produit une séquence périodique.

Dans (Palopoli et al., 2002), les périodes et les gains d'un ensemble de retours d'état connectés à des systèmes du premier ordre sont obtenus par l'optimisation d'un critère de robustesse, nommé rayon de stabilité, sous contrainte d'ordonnançabilité.

Les travaux présentés précédemment reposent sur une optimisation hors ligne. D'autres travaux portent sur une réaction en ligne. Ainsi les résultats de Eker et al. (2000) permettent d'adapter la période des commandes au taux d'occupation du processeur. L'optimisation de

la période de plusieurs commandes LQ est résolue analytiquement par l'usage d'une fonction de coût linéaire ou quadratique en la période. L'auteur montre qu'une simple dilatation des périodes nominales, en fonction de l'utilisation souhaitée du processeur, conserve l'optimalité de l'ordonnancement. Cela permet d'adapter, en ligne, les périodes à l'usage du processeur tout en optimisant la performance des commandes.

Dans (Zhao and Zheng, 1999), le processeur est attribué à la commande dont l'erreur est la plus importante pendant que les autres commandes ne sont plus calculées. (Martí et al., 2004) présente l'idée d'une allocation dynamique des ressources basée sur l'état du procédé mais le lien entre l'état et la fonction de coût est très simplifié.

Dans (Henriksson and Cervin, 2005), les auteurs optimisent en ligne la période de plusieurs correcteurs LQ. Le principe est de minimiser sur un temps fini, un critère composé des fonctions de coût utilisées pour la synthèse des commandes et de l'état courant du procédé. La complexité de l'optimisation restreint la méthode à des systèmes simples et oblige d'approcher la fonction de coût.

1.3 Contribution de la thèse

Dans le contexte que nous venons d'introduire, nous nous intéressons plus particulièrement à rendre un système de contrôle/commande plus flexible vis à vis des fluctuations des ressources de calcul.

Contribution à l'ordonnancement régulé

Le deuxième chapitre présente une régulation de la charge d'un processeur induite par un ensemble de tâches périodiques. L'approche repose sur une modélisation et un régulation inspirée des outils de la communauté automaticienne. Cette régulation va notamment adapter la période des tâches.

Synthèse d'un correcteur dynamique à période variable

Les chapitres 3 et 4 sont consacrés au développement d'une loi de commande à période variable. En effet les approches de conception conjointe commande/ordonnancement ou d'adaptation en ligne nécessitent souvent d'agir sur la période des tâches. Dans une application de commande, le correcteur doit être ajusté en fonction de la période. Peu de travaux portent sur la synthèse de correcteurs dynamiques à période variable. Nous y consacrerons la majeure partie de ce document. Le chapitre 3 est une introduction à la commande robuste linéaire qui sera appliquée à la synthèse d'un correcteur à période variable dans le chapitre 4.

Application d'un correcteur à période variable au contrôle d'un pendule T

Le chapitre 5 est consacré à l'expérimentation d'une commande à période variable présentée dans le chapitre 4 sur un procédé expérimental. Ce dernier est un pendule instable en forme de "T".

Vers l'interaction commande/ordonnancement

Dans le dernier chapitre, l'ordonnancement régulé et la commande à période variable sont regroupés dans une application commune. Cette exemple illustre la flexibilité apportée vis à vis des variations de ressources de calcul mais montre aussi l'insensibilité face à la fluctuation de la qualité de la commande. Dans un second exemple plus prospectif, une structure d'ordonnancement s'adaptant à la qualité de la commande est proposée.

1.3.1 Publication

Le travail présenté dans ce document a donnée lieu à plusieurs publications :

- Synthesis of a Sampling Period Dependent Controller using LPV Approach, D. Robert, O. Sename et D. Simon, *Proceedings of the 5th IFAC Symposium on Robust Control Design*, Toulouse, France, July, 2006
- Sampling Period Dependent RST Controller used in Control/Scheduling co-Design, D. Robert, O. Sename et D. Simon, *Proc of the 16th IFAC World Congress*, Praha, Czech Republic , July, 2005
- Robust control/scheduling co-design : application to robot control, D. Simon, D. Robert et O. Sename, *Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium*, San Francisco, United States, March, 2005
- Feedback scheduling for real-time control of systems with communication delays, O. Sename, D. Simon et D. Robert, *Proceedings of the 2003 IEEE Conference on Emerging Technologies and Factory Automation*, Lisbon, Portugal, 2003
- Real-time and delay-dependent control co-design through feedback scheduling, D. Simon, O. Sename, D. Robert et O. Testa, *Proceedings of the 15th Euromicro Conference on Real-Time Systems*, Porto, Portugal, 2003
- Conception conjointe commande/ordonnancement et ordonnancement régulé, D. Simon, O. Sename, D. Robert, chapitre 3 du livre *Systèmes Temps réel 2, ordonnancement, réseaux et qualité de service*, Hermes, 2006

Chapitre 2

Contribution à l'ordonnancement régulé

Nous proposons dans ce chapitre une contribution à la régulation de l'ordonnancement d'un ensemble de tâches périodiques. Notre approche repose sur la formulation d'un problème d'automatique. Après l'introduction de l'ordonnancement régulé, nous présentons la modélisation de l'ordonnancement sous la forme d'un système dynamique puis la synthèse du régulateur. La méthode est ensuite illustrée sur un exemple.

2.1 Introduction à l'ordonnancement adaptatif

La conception d'un ordonnancement repose toujours sur la connaissance du temps d'exécution des tâches. Sa mesure exacte est laborieuse car elle dépend de l'algorithme et de l'architecture matérielle. Les processeurs, toujours plus complexes, comportent plusieurs niveaux de cache qui réduisent le déterminisme et la reproductivité des mesures. On se contente souvent d'une borne supérieure : le temps d'exécution dans le pire cas (ou WCET). Une estimation au plus juste de cette borne est difficile et par facilité ou par prudence on la surestime. Il en résulte un sur-dimensionnement de l'ordonnancement et une sous-utilisation du processeur.

Dans le cas d'activités de durée variable, la conception de l'ordonnancement est basée sur le temps d'exécution maximal de chaque tâche. Suivant l'amplitude des variations et la distribution des durées, le sur-dimensionnement du calculateur peut être important.

Pour les systèmes critiques, où l'on parle de temps-réel "dur", le sur-dimensionnement du calculateur est justifié car les échéances doivent être strictement respectées. Tous les systèmes ne sont pas critiques et il existe un grand nombre d'applications où la contrainte temps réel dur peut être relâchée : certaines commandes, les applications multimédias, la téléphonie, ... La conséquence d'un dépassement d'échéance se traduit dans ce cas par une dégradation de la qualité : mauvais suivi de trajectoire, images saccadées ou pixélisées, mauvaise réception du signal sonore.

Le principe de contre réaction a pour avantage de permettre le rejet des perturbations et de réduire la sensibilité aux incertitudes. Dans un contexte où les contraintes temps réel peuvent être assouplies, la régulation de l'ordonnancement apporte d'une part une robustesse vis à vis des incertitudes sur les durées d'exécution des tâches et d'autre part une adaptation à la variation de ces durées ou à l'apparition d'autres tâches (perturbation). L'intérêt de cette démarche est de réduire d'une part le conservatisme dû à un dimenssionnement basé sur le temps d'exécution maximal et d'autre part la rigidité d'un ordonnancement pré-défini. Enfin la modification de l'ordonnancement en ligne ouvre de nouvelles possibilités pour l'adapter en fonction d'objectifs propres à l'application (qualité de la commande, des images, ou du son) qui peuvent évoluer au cours du temps.

La structure générale de l'adaptation est d'agir sur les paramètres des tâches (période, gestionnaire d'admission, priorité, durée d'exécution de l'algorithme, variante d'algorithme, ...) à partir des mesures de données d'ordonnancement (période, durée d'exécution, temps de réponse, nombre d'échéances dépassées, charge processeur, ...) ou d'informations propres à l'application (erreur de poursuite, niveau de bruit, nombre d'images par seconde, ...).

2.1.1 Etat de l'art

Nous présentons maintenant quelques travaux importants sur l'ordonnancement adaptatif.

Stankovic et al. (1999) présentent l'algorithme (FC-EDF) qui ajoute une contre réaction à un ordonnancement EDF. Un correcteur PID régule le taux d'échéances ratées d'un ensemble de tâches de durée variable. Pour cela, le correcteur mesure le taux d'échéances ratées sur une fenêtre de temps et agit sur le temps d'exécution des algorithmes en choisissant des variantes de durée différente. Cette méthode est étendue dans (Lu et al., 2000, 2002) où un second PID, combiné au premier, régule la charge processeur à partir de sa mesure.

Abeni et al. (2002) ajoutent une contre réaction à un serveur de tâches CBS (constant bandwidth server) introduit dans (Abeni and Buttazzo, 1998). Un serveur CBS, assimilable à un processeur virtuel doté d'une portion (bande passante) d'un processeur réel, permet d'y confiner un dépassement d'échéance. La bande passante est normalement fixée à la conception du système à partir de la connaissance des temps d'exécution. Afin de compenser les dépassements d'échéances dus aux variations du temps d'exécution, Abeni et al. (2002); Palopoli et al. (2003) utilisent un correcteur PI hybride qui modifie la bande passante en fonction du dépassement d'échéance mesuré.

Buttazzo et al. (1998) introduisent un modèle de tâche élastique applicable aux tâches périodiques. Chaque tâche est munie d'un coefficient d'élasticité et d'une période minimale et maximale. En réponse à la variation du temps d'exécution des tâches, qui est mesurée, un algorithme heuristique en modifie la période pour attribuer la charge processeur au prorata des coefficients d'élasticité, sous contraintes des périodes admissibles.

Les travaux précédemment cités ont pour unique objectif le régulation de paramètres d'ordonnancement, plus exactement la charge processeur ou le nombre d'échéances ratées. Dans le contexte de notre travail, il existe des approches spécifiques aux systèmes de contrôle/commmande dont l'objectif est d'adapter l'ordonnancement pour optimiser la commande tout en évitant la surcharge du processeur.

Shin and Meissner (1999) proposent un ordonnancement adaptatif multiprocesseur d'un ensemble de contrôleurs. Chaque contrôleur est muni d'une fonction de coût qui quantifie la qualité de sa commande en fonction de sa période. Un heuristique ajuste la période des contrôleurs et les distribue sur les processeurs en réponse à la défaillance ou à la variation de l'usage d'un processeur, le tout en optimisant la qualité globale des commandes.

Cervin et al. (2002) proposent une régulation de l'ordonnancement de plusieurs lois de commande. La période des lois de commande est modifiée en réponse à la variation de la charge du processeur. La conception du régulateur repose sur le travail de (Eker et al., 2000) qui formulent un problème d'optimisation dont l'objectif est de trouver les périodes optimales d'un ensemble de commande, chacune munie d'une fonction de coût dépendant de sa période, sous contrainte d'une charge processeur maximale.

Dans (Caccamo et al., 2000) un ensemble de lois de commande est ordonnancé via l'usage d'une variante du serveur de tâches CBS (Abeni and Buttazzo, 1998). Les périodes nominales des lois de commande sont obtenues par l'optimisation présentée dans (Seto et al., 1996) dont l'objectif est de maximiser la qualité des commandes sous contrainte d'ordonnancement. En ligne, la variation du temps d'exécution d'un correcteur est traitée localement par un serveur CBS sans perturber les autres lois de commande.

2.2 Approche proposée

Nous proposons dans ce paragraphe la régulation de la charge processeur d'un ensemble de tâches périodiques. Le schéma bloc de la figure 2.1 illustre la structure proposée.

Le régulateur d'ordonnancement estime la charge processeur, la compare à la référence U_r et modifie la fréquence f_i des tâches périodiques contrôlables. La charge processeur induite par des tâches que le régulateur ne peut pas modifier est assimilée à la perturbation en sortie U_{autres}.

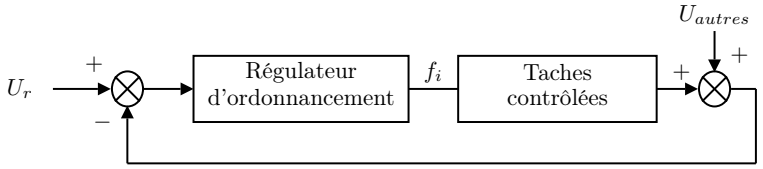


FIG. 2.1 – Boucle de régulation de l'ordonnancement

Remarque 1 *Nous nous plaçons dans un cadre restrictif où les tâches sont périodiques. Dans ce cas la charge processeur d'une tâche est estimée à partir de sa période et de la mesure de son temps d'exécution. Dans un travail préliminaire (Sename et al., 2003) nous avons estimé la charge à partir du temps de réponse, qui en pratique est plus facile à mesurer. Cette approche trop conservatrice a été abandonnée au profit de la mesure du temps d'exécution.*

L'ordonnancement utilisé est préemptif à priorités fixes. Le régulateur d'ordonnancement est une tâche autonome dont la priorité est la plus haute, ceci pour ne jamais être préemptée par une tâche qu'il contrôle. Les autres tâches ont des priorités inférieures attribuées selon un critère d'urgence relative propre à l'application. Le choix des priorités selon Rate-Monotonic ou Deadline-Monotonic est inapplicable ici car les périodes des tâches varient à l'exécution, ce qui rend impossible la vérification à priori du respect des échéances.

La période du régulateur d'ordonnancement est un degré de liberté de l'approche. Néanmoins elle doit être suffisamment grande pour permettre une bonne estimation du temps d'exécution des autres tâches. Il est même logique qu'elle soit plus grande que l'hyperpériode des tâches contrôlées, définie comme le plus petit commun multiple de leur période.

A l'exécution, les tâches sont triées dans des files d'attente ordonnées par priorité et le calculateur exécute la tâche prête dont la priorité est la plus haute. Ceci est l'ordonnancement préemptif à priorités fixes traditionnel. Il n'y a pas d'interaction directe entre l'ordonnanceur et le régulateur d'ordonnancement puisque ce dernier est une tâche comme une autre, mais dont la priorité est la plus haute et qui est capable d'obtenir le temps d'exécution et de modifier la période d'autres tâches. Ceci rend l'implémentation du régulateur d'ordonnancement facile car on ne modifie pas l'ordonnanceur du système. Il faut simplement que ce dernier fournisse le temps d'exécution des tâches et autorise la modification de leur période.

2.2.1 Modélisation de l'ordonnancement

La relation entre la période des tâches et la charge du processeur est modélisée sous la forme d'un système dynamique, à temps discret, linéaire et invariant dans le temps.

Rappelons tout d'abord que la charge processeur, pour une tâche périodique est définie par $U = \frac{c}{T}$ où c est le temps d'exécution et T la période . Dans le cas d'un système multitâche la charge processeur est la somme des charges individuelles. La charge totale est bornée entre 0 et 100 %. En utilisant la fréquence plutôt que la période la relation $U = c\,f$, avec f la fréquence, devient linéaire en f lorsque c est constant.

En pratique les mécanismes de cache du processeur et la surcharge de l'exécutif temps réel ajoutent du bruit à ces mesures. De manière similaire à Cervin et al. (2002), on filtre la mesure de charge par un filtre du premier ordre de pôle λ. Ainsi à chaque période h_s du régulateur d'ordonnancement, la charge processeur d'une tâche périodique est estimée par :

$$\hat{u}_{kh_s} = \lambda\ \hat{u}_{(k-1)h_s} + (1-\lambda)\ \bar{c}_{kh_s}\ f_{(k-1)h_s} \tag{2.1}$$

où $\bar{c}_{kh_s}$ est le temps d'exécution moyen de la tâche au cours de la période du régulateur d'ordonnancement qui vient de se terminer. Rappelons que la période de la tâche est bien inférieure à celle du régulateur d'ordonnancement. Il y a donc plusieurs exécutions de la tâche et on utilise la moyenne de leur durée.

La relation (2.1) n'est pas linéaire car $\bar{c}$ et f varient. Sachant que $\bar{c}$ est difficile à connaître au moment de la conception, on utilise, pour la synthèse du régulateur, un modèle normalisé où $\bar{c} = 1$. A l'exécution, un gain séquencé, de la forme $1/\bar{c}$ corrige la variation de la durée d'exécution par rapport au modèle normalisé, comme l'illustre la figure 2.2. La relation entre la fréquence et la charge processeur d'une tâche s'exprime alors par :

$$G_i(z) = \frac{\hat{u}_i(z)}{f_i(z)} = \frac{1-\lambda_i}{z-\lambda_i} \tag{2.2}$$

Dans le cas de n tâches, l'ordonnancement sera modélisé par la représentation d'état à temps discret suivante :

$$G(z) : \begin{cases} x_{k+1} = Ax_k + Bf_k \\ \bar{u}_k = Cx_k \end{cases} \tag{2.3}$$

où $A = diag\{\lambda_1, \ldots, \lambda_n\}$, $B = diag\{1-\lambda_1, \ldots, 1-\lambda_n\}$. $C = [1 \ldots 1]$ pour que la sortie $\bar{u}$ soit égale à la charge processeur de l'ensemble des tâches.

Le choix de λ reste libre. Il doit être compris entre 0 et 1. Nous utilisons la valeur $\lambda_i = 0.3$, pour tout i, qui est un bon compromis entre un filtrage convenable et une réponse suffisamment rapide.

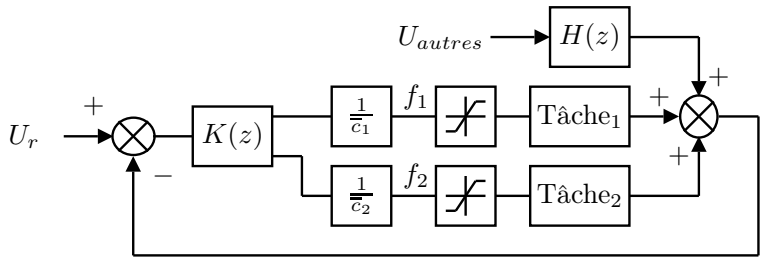


FIG. 2.2 – Schéma bloc de l'implémentation du régulateur

2.2.2 Synthèse du régulateur

Pour réguler l'ordonnancement, on utilise une commande robuste H_∞ à temps discret. Ce choix permet d'une part de garantir un minimum de robustesse, via la marge de module, et

d'autre part de contrôler un système multivariable. Pour une présentation de la commande H_∞, le lecteur peut se reporter au chapitre 3.

La formulation de la synthèse H_∞ est présentée sur la figure 2.3. Le modèle de l'ordonnancement $G(z)$, défini par l'équation (2.3), est décomposé en deux parties, $G'(z)$ et C', pour avoir en sortie de $G'(z)$ l'état complet du système, c'est à dire la mesure de charge $\hat{U}_i$ de chaque tâche. $G'(z)$ est défini par (2.3) mais avec $C = I$ et $C' = (1 \dots 1)$.

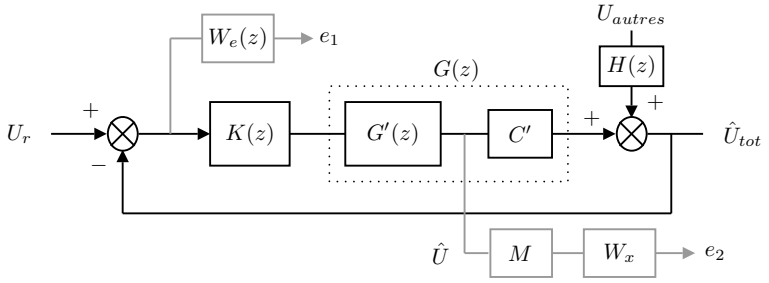


FIG. 2.3 – Formulation H_∞ du problème de commande

La charge induite par des tâches non ajustables par le régulateur est assimilée à une perturbation en sortie. La dynamique de cette perturbation est modélisée par le transfert $H(z)$. Comme nous l'avons vu dans le paragraphe de modélisation, la dynamique des tâches est modélisée par celle du filtre de la mesure. Il est alors logique que $H(z)$ soit modélisée par le filtre (2.2) car la mesure de charge est similaire.

Les objectifs de la synthèse sont décrits par les gabarits $W_e(z)$, W_x et la matrice M. $W_e(z)$ spécifie le comportement en asservissement de la boucle en réponse à un changement de référence de charge U_r. Il est usuel d'utiliser un gabarit à temps continu de la forme (2.4) que l'on discrétise à la fréquence d'échantillonnage du régulateur.

$$W_e(s) = \frac{s/M_s + \omega_s}{s + \omega_s \epsilon} \tag{2.4}$$

W_x et M définissent la répartition de charge entre les tâches. Avec $W_x = \alpha\, I_{n-1}$ où I_{n-1} est la matrice identité de dimension $n-1$ et α est un scalaire tel que $1/\alpha \approx 0$, les signaux en entrée de W_x sont maintenus proches de zéro. On peut alors forcer des relations algébriques, définies par la matrice $M \in \mathbb{R}^{(n-1)\times(n)}$, entre les charges des tâches contenues dans le vecteur $\hat{U} = (\hat{U}_1 \dots \hat{U}_n)^T$:

$$M\,\hat{U} \approx 0_{n-1} \tag{2.5}$$

où 0_{n-1} est un vecteur colonne nul de dimension $n-1$.

Dans le cas de deux tâches, pour allouer 70% de la charge U_r à la première tâche et 30% à la seconde, on écrit la relation suivante $\hat{U}_1 = 7/3\ \hat{U}_2$ qui se traduit par $M = (-1\ 7/3)$.

Dans le cas de trois tâches où la répartition est de 50%, 30% et 20%, les deux relations (2.6) s'expriment par la matrice (2.7).

$$\begin{cases} \hat{U}_1 = 5/3\ \hat{U}_2 \\ \hat{U}_2 = 3/2\ \hat{U}_3 \end{cases} \tag{2.6}$$

$$M = \begin{pmatrix} -1 & 5/3 & 0 \\ 0 & -1 & 3/2 \end{pmatrix} \tag{2.7}$$

Il suffit ensuite d'utiliser un outil de synthèse H_∞ à temps discret pour obtenir le correcteur. L'ordre de ce dernier sera égal à $n + 1$.

2.3 Exemple

Nous illustrons ici la régulation de charge sur un système composé de quatre tâches : le régulateur d'ordonnancement (reg), deux tâches périodiques (pend1 et pend2) et une tâche perturbatrice (perturb). Leurs attributs sont détaillés dans le tableau 2.1.

Tâche	Priorité	Période (ms)	Temps d'exécution (ms)
reg	1	500	1
pend1	2	4 to 400	2
perturb	3	4	2
pend2	4	4 to 400	2

TAB. 2.1 – Attributs des tâches

L'objectif est de réguler la charge processeur induite par les tâches *pend1*, *pend2* et *perturb*. Cette dernière n'est pas contrôlable et sera vue comme une perturbation. La charge du régulateur d'ordonnancement est considérée négligeable. Si tel n'est pas le cas, elle peut être assimilée à une perturbation en sortie, néanmoins cette charge étant constante, il est aussi possible de réduire d'autant la référence de charge U_r.

Seules deux tâches sont contrôlables, le modèle de l'ordonnancement est alors d'ordre 2. La valeur de λ est choisie arbitrairement à 0,3 pour les deux tâches.

Dans cette exemple, les objectifs arbitraires de la synthèse sont définis par l'équation (2.4) avec $M_s = 2$, $\omega_s = 1$ rad/s et $\epsilon = 0,05$, ce qui correspond à un temps de réponse d'environ 3 s, une erreur statique inférieure à 5 % et une bonne marge de robustesse. La répartition de charge est déterminée par $M = [-0.5\ 1]$ et $W_x = 100$, ce qui implique $U_2 - 0,5U_1 \approx 0$, c'est à dire $2U_2 \approx U_1$. Ainsi la charge allouée à la premiere tâche sera le double de celle allouée à la seconde.

Le schéma bloc de la figure 2.2 montre l'implémentation de la commande. On y retrouve la compensation de la variation du temps d'exécution via le gain $1/\bar{c}$. On y ajoute une saturation

pour maintenir la fréquence des tâches dans l'intervalle autorisé, défini dans le tableau 2.1. Notons que la saturation n'est pas prise en compte lors de la synthèse du correcteur. La présence d'un gain variable entre la sortie du correcteur et la saturation rend cette étape plus difficile.

Le correcteur obtenu est testé en simulation sur le scénario suivant. La référence de charge, initialement à 80 % diminue à 60 % à $t = 5$ s. Le temps d'exécution de la tâche *pend2* est augmenté de 20 % à $t = 9s$. Enfin la tâche perturbatrice *perturb* apparaît à $t = 12$ s. La simulation est réalisée avec la boîte à outils TrueTime (Henriksson et al., 2002) disponible pour le logiciel Matlab.

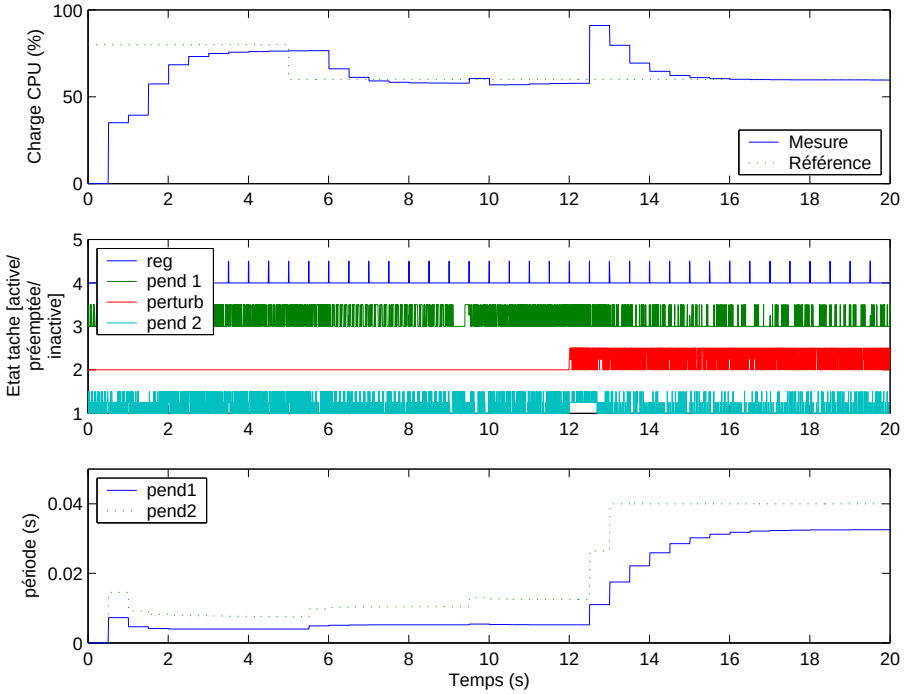


FIG. 2.4 – Evolution de la charge, chronogramme des tâches et évolution des périodes du scénario de test

Sur le premier graphique, l'évolution de la charge montre un temps de réponse de l'ordre de 3 s en réponse à une variation de la référence. On note la présence d'une erreur statique résiduelle. Les augmentations de la charge à 9 et 12 s ont été rejetées par la régulation mais avec une dynamique différente. En effet l'adaptation à une variation du temps d'exécution d'une tâche est immédiate, à une période du régulateur près, alors que la réaction face à l'apparition de la tâche *perturb* est déterminée par la dynamique de la boucle fermée.

Sur le dernier graphique, la période de *pend2* est deux fois supérieure à celle de *pend1* jusqu'à $t = 9$ s. Sachant que les deux tâches ont initialement le même temps d'exécution, il en résulte que la charge processeur de la première tâche est deux fois supérieure à celle de

la seconde, ce qui est conforme à l'objectif spécifié lors de la synthèse. Au delà de $t = 9$ s le rapport entre les deux périodes est modifié car le temps d'exécution de *pend2* augmente de 20 %, néanmoins, la répartition de charge est conservée. A partir de 13 s, cette répartition n'est plus respectée car la période de pend2 atteint sa valeur maximale 40 ms. Pour respecter la référence de charge, la période de pend1 est augmentée elle aussi. Notons cependant que la valeur de la période de pend2 en amont de la saturation est supérieure à la valeur maximale. N'ayant pris aucune précaution avec les saturations, ce phénomène est tout à fait logique. Il est alors possible, si la charge du processeur subit de grandes variations, que la régulation devienne instable.

Le deuxième graphique montre les chronogrammes d'exécution des tâches. Lors de l'apparition de la tâche perturbatrice, on voit apparaître une période transitoire où la tâche *pend2* n'est jamais inactive ce qui traduit une charge processeur assez forte sans pour autant indiquer si la tâche dépasse ou non ses échéances. Enfin la densité de chaque tracé est cohérent avec la variation de la période des tâches.

Conclusion du chapitre

Nous venons de présenter des résultats préliminaires sur la régulation de la charge processeur induite par l'ordonnancement d'un ensemble de tâches périodiques. La solution repose sur la modélisation de la relation entre la période et la charge d'une tâche sous la forme d'un système dynamique à temps discret. L'usage d'une synthèse H_∞ permet d'obtenir un correcteur dynamique et il est alors possible de définir le comportement de l'ordonnancement en régulation et en asservissement. L'ajout d'un gain séquencé permet en plus de s'adapter à la variation du temps d'exécution des tâches. L'illustration sur un exemple a montré la faisabilité de l'approche.

Une autre application a été présentée dans l'article (Simon et al., 2005) disponible en annexe. Il s'agit du contrôle d'un bras manipulateur sous contraintes de ressources de calcul. La loi de commande non linéaire est décomposée en plusieurs parties. Chacune est calculée par une tâche dédiée de période variable. L'étude de l'influence de chaque tâche sur le suivi d'une trajectoire de référence, en fonction de sa période, permet d'établir une hiérarchie d'importance entre les tâches. On en déduit alors une répartition de charge. Une application illustre l'adaptation de l'ordonnancement et de la commande à une variation de la référence de charge.

Chapitre 3

Commande H_∞/LPV

Ce chapitre introduit la commande robuste des systèmes linéaires et des systèmes linéaires à paramètres variants. Ces outils seront utilisés dans le prochain chapitre pour concevoir un correcteur à période variable.

Nous présentons tout d'abord le strict minimum pour comprendre "l'outil" qu'est la commande H_∞ des systèmes linéaires invariants dans le temps (LTI). Le lecteur curieux pourra se reporter aux nombreux livres, cours ou publications et notamment (Zhou et al., 1996) pour les détails théoriques et (Skogestad and Postlethwaite, 1996) pour son usage en pratique. Dans une seconde partie nous détaillons la commande robuste des systèmes linéaires à paramètres variants (LPV) utilisée dans la suite de ce document.

3.1 Introduction à la commande robuste H_∞

3.1.1 Quelques définitions

Un système dynamique linéaire stationnaire (dit LTI), *à temps continu*, peut être décrit par une représentation d'état :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{3.1}$$

où $x \in \mathbb{R}^{n_s}$ est le vecteur d'état, $u \in \mathbb{R}^{n_u}$ le vecteur des entrées et $y \in \mathbb{R}^{n_y}$ le vecteur des sorties. La matrice de transfert correspondante est définie par :

$$G(p) = D + C(pI - A)^{-1}B \tag{3.2}$$

Un système LTI *à temps discret* peut être décrit par une représentation d'état :

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k + Du_k \end{cases} \tag{3.3}$$

De même la matrice de transfert correspondante est définie par :

$$G(z) = D + C(zI - A)^{-1}B \tag{3.4}$$

Un système linéaire à paramètre variant (dit LPV), *à temps continu*, peut être décrit par une représentation d'état de la forme :

$$\begin{cases} \dot{x}(t) = A(\theta(t))\, x(t) + B(\theta(t))\, u(t) \\ y(t) = C(\theta(t))\, x(t) + D(\theta(t))\, u(t) \end{cases} \tag{3.5}$$

où le paramètre $\theta(t)$ évolue dans un ensemble $\Theta \subset \mathbb{R}^m$. Dans ce cas la matrice de transfert n'est pas définie. Les systèmes LPV *à temps discret* sont décrits de façon similaire par :

$$\mathcal{G} : \begin{cases} x_{k+1} = A(\theta_k)x_k + B(\theta_k)u_k \\ y_k = C(\theta_k)x_k + D(\theta_k)u_k \end{cases} \tag{3.6}$$

3.1.2 Quantification des performances

Pour étudier les systèmes dynamiques, il est nécessaire de quantifier leur performance, pour cela on utilise des normes. Dans le contexte de la commande robuste linéaire, il est usuel d'utiliser la norme H_∞ dont on rappelle la définition.

La norme H_∞ d'un système linéaire à temps continu, stable, dont la matrice de transfert est $G(p)$, est définie par :

$$\|G(p)\|_\infty = \sup_\omega \bar{\sigma}(G(j\omega)) \tag{3.7}$$

où $\bar{\sigma}$ est la valeur singulière maximale. Cette norme s'interpréte comme le gain maximal du transfert $G(p)$ sur toute la plage de fréquences et quelles que soient les directions des signaux d'entrée et de sortie. Dans le cas simple d'un système mono-entrée mono-sortie (SISO) la norme H_∞ est le maximum du tracé de Bode de $G(p)$.

La norme H_∞ d'un système linéaire à temps discret, stable, dont la matrice de transfert est $G(z)$, est définie par :

$$\|G(z)\|_\infty = \sup_{\omega \in [0,2\pi]} \bar{\sigma}(G(e^{j\omega})) \tag{3.8}$$

Dans le cas des systèmes LPV, la fonction de transfert n'est plus définie, il en est de même pour la norme H_∞. On parle alors de norme $\mathcal{L}_2$-induite définie par :

$$\|\mathcal{G}\|_\infty := \max_{u\in\mathcal{L}_2, u\neq 0} \frac{\|y\|_2}{\|u\|_2} \tag{3.9}$$

où u et y sont des signaux de carré sommable et $\|u\|_2$ est la norme $\mathcal{L}_2$ du signal u définie par :

$$\|u(t)\|_2 = \sqrt{\int_{-\infty}^{+\infty} u(\tau)^2 d\tau} \qquad \text{pour un signal à temps continu} \tag{3.10}$$

$$\|u(k)\|_2 = \sqrt{\sum_{i=-\infty}^{+\infty} u(i)^2} \qquad \text{pour un signal à temps discret} \tag{3.11}$$

La norme $\mathcal{L}_2$-induite, quand les matrices de la représentation d'état sont constantes, est équivalente à la norme H_∞. Notons que si θ est fixé alors la représentation d'état (3.6) d'un système LPV est identique à celle d'un système LTI.

Remarque 2 *L'absence d'une fonction de transfert implique qu'il n'existe pas de tracé de Bode pour un système LPV. Toutes les analyses fréquentielles que nous ferons par la suite ne sont valables que pour une valeur de θ fixée. Néanmoins un maillage de l'ensemble Θ permet d'estimer le comportement fréquentiel du système pour de "nombreuses" valeurs de θ.*

3.1.3 Spécification des objectifs

Soit le système en boucle fermée de la figure 3.1. Une approche classique pour concevoir le régulateur K est d'étudier le transfert de la chaîne directe $L = GK$. Ainsi, en étudiant ses réponses en fréquence et en phase, on peut agir sur des caractéristiques telles que les marges de gain et de phase, la bande passante du correcteur, la présence d'intégrateurs ou de retards.

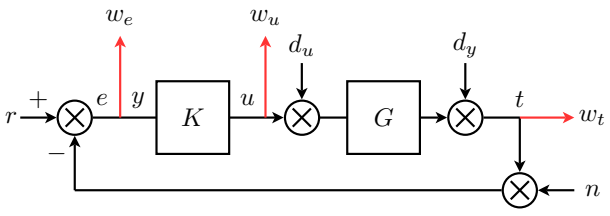


FIG. 3.1 – Exemple d'interconnexion

Cette méthodologie ne permet pas de spécifier directement des transferts de la boucle fermée. Pour cela, une autre formulation repose sur le problème standard de la figure 3.2 où les objectifs sont définis sur le transfert entre l'entrée w et la sorties z.

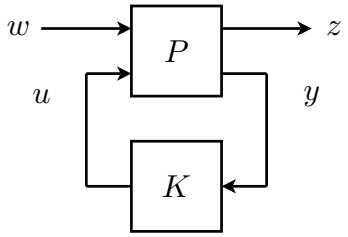


FIG. 3.2 – Problème standard

Le problème standard est une forme très générale facile à obtenir. Prenons l'exemple de l'interconnexion de la figure 3.1 avec les entrées exogènes r, d_u, d_y, n, les sorties exogènes w_e, w_u, w_t l'entrée de commande u et la sortie mesurée y. Le problème standard correspondant a pour entrée le vecteur $w = (r, d_u, d_y, n)^T$ et pour sortie le vecteur $z = (w_e, w_u, w_t)^T$. Ainsi le transfert multi-variables T_{wz} contient les différentes fonctions de sensibilité de la boucle fermée et notamment :

- le suivi de trajectoire via la relation e/r,
- la relation entrée-sortie t/r,
- la sensibilité de la commande aux bruits u/n et aux perturbations u/d_u et u/d_y,
- le rejet de perturbations via e/d_u, e/d_y, t/d_u, t/d_y.

L'extension du problème standard par l'ajout des gabarits fréquentiels W_I et W_O, illustré sur la figure 3.3, permet d'agir sur les différentes fonctions de sensibilité.

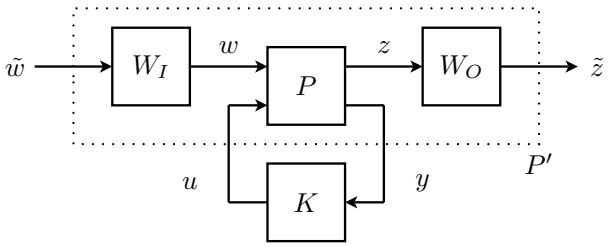


FIG. 3.3 – Problème standard étendu

Prenons l'exemple de la figure 3.1 et notamment le transfert entre e et r représenté ici par la fonction de sensibilité suivante :

$$S(p) = \frac{1}{1 + G(p)K(p)} \tag{3.12}$$

On peut agir sur le suivi de trajectoire en imposant une forme à ce transfert via un gabarit fréquentiel $W_S(p)$. Il est usuel pour cette fonction de sensibilité d'utiliser un gabarit

similaire à l'expression (3.13) dont le tracé est présenté sur la figure 3.4. Dans ce cas, ϵ_S définit l'erreur statique maximale, ω_S influence la rapidité du système et M_S borne la marge de module.

$$W_S(p) = \frac{p/M_S + \omega_S}{p + \omega_S \epsilon_S} \tag{3.13}$$

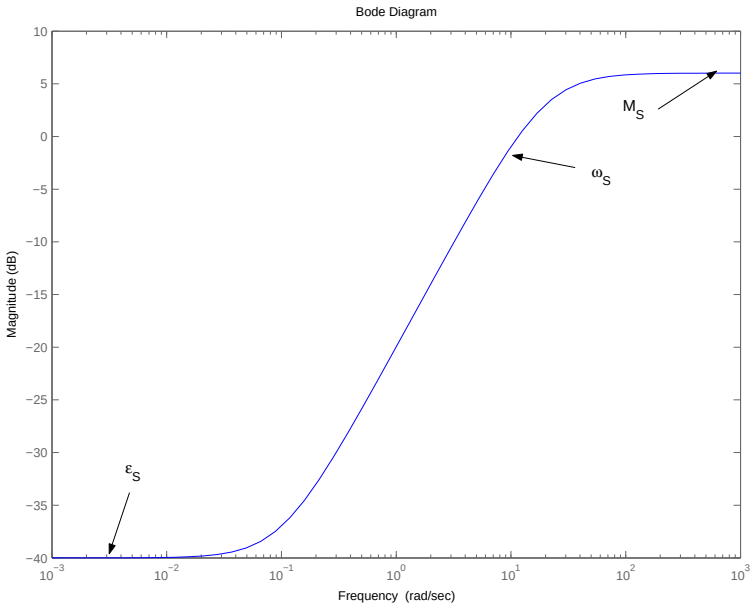


FIG. 3.4 – Réponse fréquentielle de $1/W_S(p)$ avec $\epsilon_S = 0.01$, $\omega_S = 10$ rad/s et $M_S = 2$

Pour garantir une fonction de sensibilité $S(p)$ bornée par son gabarit, il faut vérifier $|S(p)| < |1/W_S(p)|, \forall p$. Cette condition est équivalente à $\|W_S(p)S(p)\|_\infty \leq 1$. Sur cette exemple, en posant $z = e$, $w = r$, $W_I = I$ et $W_O(p) = W_S(p)$ selon la figure 3.3, l'objectif de la synthèse H_∞ est de trouver un correcteur linéaire $K(p)$ tel que le système en boucle fermée soit stable de manière interne et que le transfert $\bar{z}/\bar{w}$ ait une norme H_∞ inférieure à 1, c'est à dire $\|W_S(p)S(p)\|_\infty \leq 1$.

Il est possible d'agir simultanément sur plusieurs fonctions de sensibilité. Dans l'exemple du problème de sensibilité mixte de l'équation (3.14), les fonctions de sensibilité S et KS doivent respecter les gabarits fréquentiels $W_S(p)$ et $W_{KS}(p)$. Dans ce cas l'inégalité (3.14) est une condition suffisance, mais non nécessaire, pour que chaque fonction respecte son gabarit.

$$\left\| \begin{array}{c} W_S(p)\, S(p) \\ W_{KS}(p)\, KS(p) \end{array} \right\|_\infty \leq 1 \tag{3.14}$$

Par la suite on parlera de problème standard étendu incluant la spécification des objectifs. Le problème est dit étendu car le vecteur d'état de P' contient l'état de P étendu des états des gabarits fréquentiels W_I et W_O.

Nous ne détaillerons pas ici les conditions d'existence d'un correcteur K solution du problème standard étendu dans le cas des systèmes LTI. Pour cela, le lecteur peut se référer notamment à (Zhou et al., 1996). La section suivante est consacrée aux systèmes linéaires à paramètres variants et notamment à la recherche d'un correcteur K qui répond aux mêmes exigences.

3.2 Commande des systèmes LPV polytopiques

Les récents travaux sur les systèmes LPV ont été motivés par le contrôle d'un système non linéaire à partir d'outils de la commande des systèmes linéaires.

L'approche traditionnellement utilisée est le séquencement de gains qui améliore la stabilité et la performance par rapport à l'usage d'un correcteur linéaire unique. Elle consiste à modéliser un système non-linéaire par un ensemble de modèles linéaires, valables autour de différents points de fonctionnement. Un correcteur LTI est conçu pour chaque modèle et une loi globale de séquencement ou d'interpolation, dépendant des paramètres mesurables en ligne, définit les transitions entre correcteurs.

Cette approche est attractive car elle utilise les outils de la commande linéaire, ce qui facilite la modélisation et la synthèse des correcteurs. Néanmoins la mise au point doit être répétée pour chaque correcteur, dont le nombre peut être élevé, et la loi de séquencement ou d'interpolation peut être difficile à trouver. Enfin il n'y a pas de garantie théorique de la stabilité et de la performance de la boucle fermée quelles que soient les transitions entre correcteurs (Shamma and Athans, 1991).

Les travaux sur le contrôle des systèmes LPV ont été inspirés par les résultats d'analyse et de synthèse de la commande robuste des systèmes LTI. On peut les regrouper en deux familles.

La première est basée sur une extension du théorème du petit gain. Elle a été introduite par Packard (1994) et Apkarian et al. (1995). La dépendance dans les paramètres variants du procédé et du correcteur s'exprime sous une forme linéaire fractionnelle (LFT) et l'existence d'un correcteur est caractérisée par un ensemble d'inégalités matricielles linéaires (LMI). Les résultats initiaux ont été étendus pour en réduire le conservatisme notamment par Scorletti and El Ghaoui (1998) et Scherer (2001).

La seconde famille repose sur l'usage d'une fonction de Lyapunov quadratique en l'état. Elle a été introduite par Becker and Packard (1994) et Apkarian et al. (1995). L'existence et la caractérisation du correcteur est formulée par des inégalités matricielles linéaires. Ces résultats ont été étendus pour tenir compte de la vitesse de variation du paramètre ou utiliser une fonction de Lyapunov dépendante des paramètres variants notamment dans Apkarian and Adams (1998).

Pour plus de détails, le lecteur peut se référer à l'article de Wu (2001) et à ses références.

Les travaux qui viennent d'être cités sont majoritairement dédiés aux systèmes LPV à temps continu. Notre problématique, la conception d'un correcteur à période variable, utilise une formulation à temps discret. Dans la suite de ce chapitre nous rappelons brièvement des résultats de la littérature consacrés aux systèmes LPV à temps discret. Ces résultats sont issus des articles de Gahinet and Apkarian (1994), Apkarian et al. (1995), de Oliveira et al. (1999), et Oliveira et al. (2002). Ces travaux se basent sur une classe particulière de systèmes, les systèmes LPV polytopiques, issus de la représentation d'incertitudes sous forme de polytope pour la commande robuste.

3.2.1 Modélisation

Un système LPV, à temps discret, dit "polytopique" est défini par le modèle suivant :

$$\begin{cases} x_{k+1} = A(\theta_k)x_k + B(\theta_k)u_k \\ y_k = C(\theta_k)x_k + D(\theta_k)u_k \end{cases} \tag{3.15}$$

où $A(\theta_k)$, $B(\theta_k)$, $C(\theta_k)$, $D(\theta_k)$ sont affines en θ_k et θ_k évolue dans un polytope Θ de sommets $\omega_1, \ldots, \omega_N$, i.e. :

$$\theta_k \in \Theta := \mathbf{Co}\{\omega_i, \ldots, \omega_N\} \tag{3.16}$$

θ_k s'exprime comme la combinaison convexe des N vecteurs w_i :

$$\theta_k = \sum_{i=1}^{N} \alpha_i(k)\, \omega_i, \quad \text{avec} \quad \alpha_i(k) \geq 0, \sum_{i=1}^{N} \alpha_i(k) = 1 \tag{3.17}$$

Les matrices d'état étant affines en θ_k, elles s'expriment comme la combinaison convexe de N matrices sommets, images des sommets ω_i par $A(w_i)$, $B(w_i)$, $C(w_i)$ et $D(w_i)$:

$$\begin{pmatrix} A(\theta_k) & B(\theta_k) \\ C(\theta_k) & D(\theta_k) \end{pmatrix} = \sum_{i=1}^{N} \alpha_i(k) \begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix} = \sum_{i=1}^{N} \alpha_i(k) \begin{pmatrix} A(w_i) & B(w_i) \\ C(w_i) & D(w_i) \end{pmatrix}, \tag{3.18}$$

Les N systèmes sommets, définis par des matrices d'état constantes, et les coordonnées polytopiques α_i sont suffisants pour décrire le système.

Transformation d'un modèle affine en modèle polytopique

Ecrire le modèle d'un système LPV directement sous forme polytopique est difficile car les paramètres variants d'un système physique s'expriment rarement comme une combinaison convexe. Il est plus naturel d'avoir une représentation d'état affine en un vecteur de paramètres $\beta_k = (\beta_1(k), \ldots, \beta_n(k))$:

$$\begin{pmatrix} A(\beta_k) & B(\beta_k) \\ C(\beta_k) & D(\beta_k) \end{pmatrix} = \sum_{j=1}^{n} \beta_j(k) \begin{pmatrix} A_j & B_j \\ C_j & D_j \end{pmatrix} \tag{3.19}$$

où chaque composante $\beta_j(k)$ est bornée par $[\underline{\beta}_j; \overline{\beta}_j]$.

Cette forme affine se transforme aisément en un modèle polytopique. En effet les composantes $\beta_j(k)$ étant bornées, β_k prend des valeurs dans un hypercube Θ de 2^n sommets. Chaque sommet de ce polytope est défini par un vecteur $\omega_i = (\nu_{i1}, \ldots, \nu_{in})$ où ν_{ij} prend l'une des deux valeurs $\{\underline{\beta}_j, \overline{\beta}_j\}$.

La représentation d'état de chaque système sommet, c'est à dire les matrices A_i, B_i, C_i et D_i de l'équation (3.18), est obtenue par l'évaluation de l'expression (3.19) pour $\beta_k = \omega_i$.

L'expression des coordonnées polytopiques $\alpha_i(k)$ en fonction des paramètres $\beta_j(k)$ nécessite la résolution du système d'équations suivant :

$$\begin{cases} \displaystyle\sum_{i=1}^{2^n} \alpha_i(k)\nu_{ij} = \beta_j(k) \qquad j = 1 \ldots n \\ \displaystyle\sum_{i=1}^{2^n} \alpha_i(k) = 1, \ \alpha_i(k) \geq 0 \end{cases} \tag{3.20}$$

Ce système est sous dimensionné car il possède $n+1$ équations et 2^n inconnues. Pour le résoudre nous utilisons l'algorithme présent dans la boîte à outils LMI de Matlab (Gahinet et al., 1996).

Soit un système affine en un vecteur de paramètres $\beta = (\beta_1, \ldots, \beta_n)$ où chaque paramètre β_j appartient à l'intervalle $[\underline{\beta}_j; \overline{\beta}_j]$ et la dépendance temporelle est supprimée pour alléger l'écriture. β évolue dans un polytope formé de 2^n sommets $\omega_i = (\nu_{i1}, \ldots, \nu_{in})$ où ν_{ij} prend l'une des deux valeurs $\{\underline{\beta}_j, \overline{\beta}_j\}$. Les coordonnées polytopiques sont alors définies par :

$$\alpha_i = \prod_{j=1}^{n} \tau_j \quad i = 1 \ldots 2^n \tag{3.21}$$

$$\text{avec} \begin{cases} \tau_j = \dfrac{\beta_j - \underline{\beta}_j}{\overline{\beta}_j - \underline{\beta}_j} & \text{si la } j^{\text{ème}} \text{ composante de } \omega_i \text{ est } \overline{\beta}_j \\ \tau_j = \dfrac{\overline{\beta}_j - \beta_j}{\overline{\beta}_j - \underline{\beta}_j} & \text{si la } j^{\text{ème}} \text{ composante de } \omega_i \text{ est } \underline{\beta}_j \end{cases} \tag{3.22}$$

Exemple Prenons l'exemple d'un système affine en deux paramètres a et b. Ce système peut être mis sous la forme de l'expression (3.19). Le polytope Θ, composé de 4 sommets, est illustré sur la figure 3.5. A chaque sommet de Θ correspond un système sommet $S_i, i = 1 \ldots 4$.

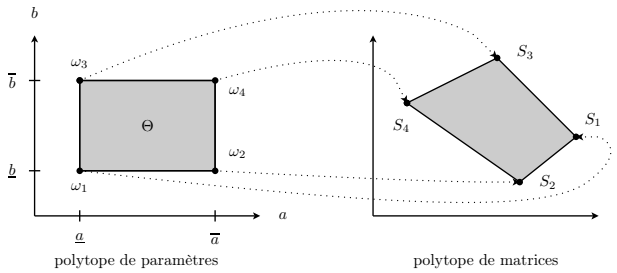


FIG. 3.5 – Illustration du polytope de paramètres Θ et du polytope de systèmes

Les coordonnées polytopiques s'expriment alors par :

$$\begin{aligned}
\omega_1 &= (\underline{a}, \underline{b}) \qquad \alpha_1 = \left(\frac{\overline{a}-a}{\overline{a}-\underline{a}}\right)\left(\frac{\overline{b}-b}{\overline{b}-\underline{b}}\right) \\
\omega_2 &= (\overline{a}, \underline{b}) \qquad \alpha_2 = \left(\frac{a-\underline{a}}{\overline{a}-\underline{a}}\right)\left(\frac{\overline{b}-b}{\overline{b}-\underline{b}}\right) \\
\omega_3 &= (\underline{a}, \overline{b}) \qquad \alpha_3 = \left(\frac{\overline{a}-a}{\overline{a}-\underline{a}}\right)\left(\frac{b-\underline{b}}{\overline{b}-\underline{b}}\right) \\
\omega_4 &= (\overline{a}, \overline{b}) \qquad \alpha_4 = \left(\frac{a-\underline{a}}{\overline{a}-\underline{a}}\right)\left(\frac{b-\underline{b}}{\overline{b}-\underline{b}}\right)
\end{aligned} \tag{3.23}$$

Il est aisé de vérifier que la somme des α_i vaut 1.

3.2.2 Intérêt du modèle polytopique

On peut s'interroger sur l'intérêt de transformer un modèle affine en un modèle polytopique. La raison est que les outils que nous utilisons par la suite reposent sur un modèle polytopique. L'avantage de ce modèle tient dans ses propriétés de convexité qui facilitent l'analyse et la synthèse de correcteurs.

En effet, l'expression des matrices d'état sous forme d'une combinaison convexe (3.18) permet, à partir de l'étude des systèmes aux sommets du polytope, d'obtenir des résultats valables pour l'ensemble du polytope. Sachant que chaque système sommet est identique à un système LTI, on peut étendre, relativement facilement, les résultats applicables aux systèmes LTI vers les systèmes LPV polytopiques. Nous illustrons cette idée sur l'analyse de la stabilité quadratique.

Définition 1 *Le système à temps discret autonome suivant :*

$$x_{k+1} = A_k\, x_k, \qquad x_k \in \mathbb{R}^n, \quad A_k \in \Omega \subseteq \mathbb{R}^{n\times n} \tag{3.24}$$

est quadratiquement stable s'il existe une fonction de Lyapunov $V(x_k) = x_k^T P x_k,\ P > 0$, *quadratique en* x_k, *telle que :*

$$V(x_{k+1}) - V(x_k) < 0, \quad \forall x_k \textit{ satisfaisant } (3.24) \textit{ avec } x_0 \textit{ non nulle} \tag{3.25}$$

Un condition nécessaire et suffisante pour la stabilité quadratique du système (3.24) est de trouver une matrice P telle que :

$$P > 0, \quad \mathbf{A}^T P \mathbf{A} - P < 0 \qquad \forall \mathbf{A} \in \Omega \tag{3.26}$$

ce qui est équivalent à trouver une matrice P qui vérifie :

$$\begin{pmatrix} -P^{-1} & \mathbf{A} \\ \mathbf{A} & -P \end{pmatrix} < 0 \qquad \forall \mathbf{A} \in \Omega \tag{3.27}$$

Dans le cas d'un système LTI, on a $A_k = A$ et il suffit de trouver une matrice P qui vérifie l'unique inégalité suivante :

$$\begin{pmatrix} -P^{-1} & A \\ A & -P \end{pmatrix} < 0 \tag{3.28}$$

Pour un système LPV, on a $A_k = A(\theta_k)$. Si la dépendance en θ_k est quelconque, il faut vérifier l'expression (3.27) pour une infinité de valeurs, ce qui est impossible. Dans le cas d'un système LPV polytopique, $A(\theta_k)$ est la combinaison convexe de N matrices A_i et il suffit de vérifier l'inégalité matricielle pour chaque A_i :

$$\begin{pmatrix} -P^{-1} & A(\theta_k) \\ A(\theta_k) & -P \end{pmatrix} < 0,\ \forall \theta_k \in \Theta \ \Leftrightarrow\ \begin{pmatrix} -P^{-1} & A_i \\ A_i & -P \end{pmatrix} < 0 \quad , i = 1, \ldots, N \tag{3.29}$$

Cette facilité à rendre un problème de dimension infinie (tester toutes les valeurs de θ_k) en un problème de dimension finie (tester seulement les systèmes aux sommets du polytope) motive l'usage d'un modèle LPV polytopique.

3.2.3 Analyse de la stabilité

Nous rappelons ici des résultats d'analyse de stabilité des systèmes LPV à temps discret qui sont à la base des résultats de synthèse de correcteurs.

Soit le système LPV polytopique discret et autonome de l'expression (3.30) où θ_k, défini par (3.17), évolue dans un polytope de N sommets.

$$x_{k+1} = A(\theta_k) x_k \tag{3.30}$$

Lemme 1 (Stabilité quadratique)
Les propositions suivantes sont équivalentes (Boyd et al., 1994; Apkarian et al., 1995) :

(i) le système (3.30) *est quadratiquement stable, pour tout* $\theta_k \in \Theta$

(ii) il existe une matrice $P = P^T > 0$ *telle que :*

$$A_i^T P A_i - P < 0, \quad i = 1 \dots N$$

(iii) il existe une matrice $P = P^T > 0$ *telle que :*

$$\begin{pmatrix} -P^{-1} & A_i \\ A_i^T & -P \end{pmatrix} < 0, \qquad i = 1 \dots N$$

(iv) il existe une matrice $P = P^T > 0$ *telle que :*

$$\begin{pmatrix} P & A_i P \\ P A_i^T & P \end{pmatrix} > 0, \qquad i = 1 \dots N$$

La stabilité est ici démontrée par l'existence d'une fonction de Lyapunov de la forme $V(x_k) = x_k^T P x_k$ où la matrice de Lyapunov P est constante. Notons que dans les expressions (ii) et (iii) la matrice P doit vérifier simultanément les N inégalités. Aucune contrainte n'est imposée à la trajectoire θ_k tant qu'elle reste dans Θ ce qui se traduit par une évolution qui peut être infiniment rapide.

Lemme 2 (Caractérisation étendue de la stabilité)
Le système (3.30) *est asymptotiquement stable, pour tout* $\theta_k \in \Theta$*, s'il existe des matrices* $P_i = P_i^T > 0$ *et une matrice* G *quelconque telle que (de Oliveira et al., 1999) :*

$$\begin{pmatrix} P_i & A_i G \\ G A_i^T & G + G^T - P_i \end{pmatrix} > 0, \qquad i = 1 \dots N$$

La stabilité est alors démontrée par l'usage d'une fonction de Lyapunov de la forme :

$$V(x_k) = x_k^T (\sum_{i=1}^{N} \alpha_i(k) P_i) x_k \tag{3.31}$$

Contrairement au lemme 1 la fonction de Lyapunov est ici dépendante des coordonnées polytopiques, donc du paramètre θ_k, ce qui permet de réduire le conservatisme par rapport à une fonction indépendante du paramètre variant.

3.2.4 Analyse de la performance

Nous rappelons ici les résultats de caractérisation, sous la forme d'une LMI, du gain $\mathcal{L}_2$-induit entre l'entrée u et la sortie y du système LPV discret, polytopique, de l'expression (3.32) où θ_k, défini par (3.17), évolue dans un polytope de N sommets.

$$\begin{cases} x_{k+1} = A(\theta_k)x_k + B(\theta_k)u_k \\ y_k = C(\theta_k)x_k + D(\theta_k)u_k \end{cases} \tag{3.32}$$

Rappelons qu'un système est stable de façon interne si toutes les valeurs propres de sa matrice A sont stables. Cela correspond à obtenir un signal de sortie borné quel que soit le point d'application d'un signal d'excitation borné. La stabilité interne est à différencier de la stabilité BIBO (entrée bornée/sortie bornée) qui implique une sortie bornée pour un signal d'excitation borné appliqué uniquement en entrée du système. La stabilité interne garantit donc la stabilité BIBO, mais la réciproque est fausse. Dans le cas d'un système LPV la stabilité interne doit être vérifiée pour toute trajectoire du paramètre variant contenue dans son domaine de définition.

Lemme 3 (Caractérisation du gain $\mathcal{L}_2$-induit)

Les propositions suivantes sont équivalentes (Apkarian et al., 1995) :

(i) La stabilité interne du système (3.32) *est garantie et le gain $\mathcal{L}_2$-induit de la relation entrée/sortie est bornée par γ, i.e. $\|y\|_2 < \gamma\|u\|_2$, pour tout $\theta_k \in \Theta$*

(ii) Il existe une matrice $P = P^T > 0$ et un scalaire γ tels que

$$\begin{pmatrix} -P^{-1} & A_i & B_i & 0 \\ A_i^T & -P & 0 & C_i^T \\ B_i^T & 0 & -\gamma I & D_i^T \\ 0 & C_i & D_i & -\gamma I \end{pmatrix} < 0, \qquad i = 1 \dots N$$

Lemme 4 (Caractérisation étendue du gain $\mathcal{L}_2$-induit)

La stabilité interne du système (3.32) *est garantie et le gain $\mathcal{L}_2$-induit de la relation entrée/sortie est bornée par γ, i.e. $\|y\|_2 < \gamma\|u\|_2$, pour tout $\theta_k \in \Theta$, s'il existe des matrices $P_i = P_i^T > 0$, une matrice G quelconque et un scalaire γ tels que (Oliveira et al., 2002) :*

$$\begin{pmatrix} P_i & A_iG & B_i & 0 \\ GA_i^T & G + G^T - P_i & 0 & G^TC_i^T \\ B_i^T & 0 & I & D_i^T \\ 0 & C_iG & D_i & \gamma^2 I \end{pmatrix} > 0, \qquad i = 1 \dots N$$

Comme dans le cas de l'analyse de stabilité, la caractérisation étendue permet d'utiliser une fonction de Lyapunov dépendante du paramètre variant dans le but de réduire le conservatisme induit par l'usage d'une fonction de Lyapunov indépendante de θ_k.

3.2.5 Synthèse de contrôleurs

Nous abordons maintenant le contrôle d'un système LPV sous forme polytopique.

Objectif

Soit un système LPV discret polytopique $P(\theta_k)$ reliant les entrées exogènes w et de contrôle u, aux sorties mesurées y et contrôlées z. Ce système est défini par la représentation d'état suivante, où la dépendance temporelle de θ, w, u, z, y est omise par soucis de clarté. Le paramètre θ, défini par l'expression (3.17), évolue dans le polytope Θ, de N sommets, défini par l'expression (3.16).

$$P(\theta) : \begin{cases} x_{k+1} = A(\theta)x_k + B_w(\theta)w + B_u(\theta)u \\ z = C_z(\theta)x_k + D_{zw}(\theta)w + D_{zu}(\theta)u \\ y = C_y(\theta)x_k + D_{yw}(\theta)w + D_{yu}(\theta)u \end{cases} \tag{3.33}$$

où $u \in \mathbb{R}^{n_u}$, $y \in \mathbb{R}^{n_y}$, $z \in \mathbb{R}^{n_z}$, $w \in \mathbb{R}^{n_w}$ et le vecteur d'état $x_k \in \mathbb{R}^{n_s}$.

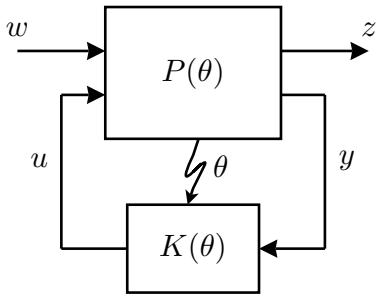


FIG. 3.6 – Interconnexion étudiée

On suppose que le vecteur de paramètres θ est mesurable en temps réel. L'objectif est alors de trouver un correcteur $K(\theta)$, séquencé en ligne par θ, de la forme :

$$K(\theta) : \begin{cases} \tilde{x}_{k+1} = A_K(\theta)\tilde{x}_k + B_K(\theta)y \\ u = C_K(\theta)\tilde{x}_k + D_K(\theta)y \end{cases} \tag{3.34}$$

où $\tilde{x}_k \in \mathbb{R}^{n_s}$, i.e. un correcteur du même ordre que le système $P(\theta)$.

Ce contrôleur doit :
- garantir la stabilité interne de l'interconnexion de la figure 3.6,
- garantir une norme $\mathscr{L}_2$-induite entre l'entrée w et la sortie z bornée par γ.

Nous présentons ici deux formulations de la littérature qui se différencient par l'usage d'une fonction de Lyapunov indépendante du paramètre θ et dépendante pour la seconde. Ces deux formulations reposent sur les hypothèses suivantes :

Hypothèses 1

(H_1) $D_{yu}(\theta) = 0$,

(H_2) les matrices $B_u(\theta), C_y(\theta), D_{zu}(\theta), D_{yw}(\theta)$ *sont indépendantes de* θ,

(H_3) les paires $(A(\theta), B_u)$ *et* $(A(\theta), C_y)$ *sont respectivement quadratiquement stabilisable et quadratiquement détectable pour toute évolution de* θ. *Ceci est équivalent à l'existence des matrices réelles* $X = X^T > 0$ *et* $Y = Y^T > 0$ *satisfaisant :*

$$\mathcal{N}_u^T [A_i^T X A_i - X] \mathcal{N}_u < 0, \quad i = 1, 2, \ldots, N \tag{3.35}$$

$$\mathcal{N}_y^T [A_i^T Y A_i - X] \mathcal{N}_y < 0, \quad i = 1, 2, \ldots, N \tag{3.36}$$

où $\mathcal{N}_u$ *et* $\mathcal{N}_y$ *sont respectivement des bases des noyaux de* B_u^T *et* C_y,

Remarque 3 *L'hypothèse* H_3 *implique que les modes non commandables et non observables du système LPV considéré soient quadratiquement stables pour toute évolution du paramètre* θ *dans son ensemble de définition. On reconnaît dans les expressions* (3.35) *et* (3.36) *le test de stabilité quadratique de l'équation* (3.26).

Fonction de Lyapunov indépendante de θ

Théorème 1 (Condition de faisabilité)

Sous les hypothèses 1, il existe un correcteur de la forme (3.34), *avec* $\tilde{x} \in \mathbb{R}^{n_s}$, *qui garantit (Apkarian et al., 1995) :*

- *la stabilité interne de la boucle fermée*
- *un gain* $\mathscr{L}_2$ *entre* w *et* z *borné par* γ, *i.e.* $\|z\|_2 < \gamma \|w\|_2$

s'il existe $\gamma \in \mathbb{R}$ *et deux matrices symétriques* $R, S \in R^{n_s \times n_s}$ *satisfaisant les* $2N+1$ *LMIs suivantes :*

$$\left(\begin{array}{c|c} \mathcal{N}_R & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i R A_i^T - R & A_i R C_{zi}^T & B_{wi} \\ C_{zi} R A_i^T & -\gamma I + C_{zi} R C_{zi}^T & D_{zwi} \\ \hline B_{wi}^T & D_{zwi}^T & -\gamma I \end{array}\right) \left(\begin{array}{c|c} \mathcal{N}_R & 0 \\ \hline 0 & I \end{array}\right) < 0, \quad i = 1 \ldots N \tag{3.37}$$

$$\left(\begin{array}{c|c} \mathcal{N}_S & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i^T S A_i - S & A_i^T S B_{wi} & C_{zi}^T \\ B_{wi}^T S A_i & -\gamma I + B_{wi}^T S B_{wi} & D_{zwi}^T \\ \hline C_{zi} & D_{zwi} & -\gamma I \end{array}\right) \left(\begin{array}{c|c} \mathcal{N}_S & 0 \\ \hline 0 & I \end{array}\right) < 0, \quad i = 1 \ldots N \tag{3.38}$$

$$\left(\begin{array}{cc} R & I \\ I & S \end{array}\right) \geq 0 \tag{3.39}$$

où A_i, B_{wi}, C_{zi}, D_{zwi} sont $A(\theta)$, $B_w(\theta)$, $C_z(\theta)$, $D_{zw}(\theta)$ évaluées au i^{th} sommet du polytope de paramètres Θ. $\mathcal{N}_S$ and $\mathcal{N}_R$ dénotent respectivement une base des noyaux de (B_u^T, D_{zu}^T) et (C_y, D_{yw}).

Remarque 4 *Ce théorème est obtenu par l'application du lemme 3 d'analyse sur la boucle fermée de la figure 3.6. Malheureusement l'inégalité matricielle obtenue n'est plus linéaire en les inconnus car des produits entre la matrice P et les matrices du correcteur apparaissent. L'objectif est alors d'obtenir une inégalité matricielle linéaire et pour cela plusieurs approches existent. Le théorème fait appel au lemme de projection (Gahinet and Apkarian, 1994) qui fait disparaître les matrices du correcteur. D'autres approches utilisent un changement de variables qui linéarise l'inégalité matricielle (Bernussou et al., 1989; Scherer et al., 1997).*

Si le théorème 1 est vérifié on peut alors reconstruire les correcteurs correspondant à chaque sommet du polytope Θ. Deux approches sont possibles, soit par la résolution d'un problème d'optimisation, soit par le calcul direct (Gahinet, 1996). Nous utilisons ici la première solution.

Reconstruction des correcteurs par optimisation A parti des matrices R et S on construit les matrices M et $N \in \mathbb{R}^{n_s}$, par une décomposition en valeurs singulières (Gahinet and Apkarian, 1994) :

$$MN^T = I - RS = U\Sigma V^T = (U\Sigma^{\frac{1}{2}})(V\Sigma^{\frac{1}{2}})^T \tag{3.40}$$

On construit alors la matrice X_{cl} qui est l'unique solution du système linéaire suivant :

$$\begin{pmatrix} S & I \\ N^T & 0 \end{pmatrix} = X_{cl} \begin{pmatrix} I & R \\ 0 & M^T \end{pmatrix} \tag{3.41}$$

Pour la reconstruction de chacun des N correcteurs sommets, caractérisé par la matrice $\mathbf{\Omega}_i$, il faut résoudre l'inégalité suivante :

$$\mathbf{\Psi}_i + P^T \mathbf{\Omega}_i^T Q + Q^T \mathbf{\Omega}_i P < 0, \qquad i = 1, \ldots, N \tag{3.42}$$

avec

$$\mathbf{\Psi}_i = \begin{pmatrix} -X_{cl}^{-1} & \mathbf{A}_i & \mathbf{B}_i & 0 \\ \mathbf{A}_i^T & -X_{cl} & 0 & \mathbf{C}_i^T \\ \mathbf{B}_i^T & 0 & -\gamma I & \mathbf{D}_i^T \\ 0 & \mathbf{C}_i & \mathbf{D}_i & -\gamma I \end{pmatrix} \qquad \mathbf{\Omega}_i = \begin{pmatrix} A_{k_i} & B_{k_i} \\ C_{k_i} & D_{k_i} \end{pmatrix} \tag{3.43}$$

$$\mathbf{A}_i = \begin{pmatrix} A_i & 0 \\ 0 & 0_k \end{pmatrix} \quad \mathbf{B}_i = \begin{pmatrix} B_{w_i} \\ 0 \end{pmatrix} \quad \mathbf{C}_i = \begin{pmatrix} C_{z_i} & 0 \end{pmatrix} \tag{3.44}$$

$$Q = \begin{pmatrix} 0 & \mathcal{C} & \mathcal{D}_{21} & 0 \end{pmatrix} \qquad P = \begin{pmatrix} \mathcal{B}^T & 0 & 0 & \mathcal{D}_{12}^T \end{pmatrix} \tag{3.45}$$

$$\mathcal{B} = \begin{pmatrix} 0 & B_u \\ I_k & 0 \end{pmatrix} \quad \mathcal{C} = \begin{pmatrix} 0 & I_k \\ C_y & 0 \end{pmatrix} \quad \mathcal{D}_{12} = \begin{pmatrix} 0 & D_{zu} \end{pmatrix} \quad \mathcal{D}_{21} = \begin{pmatrix} 0 \\ D_{yw} \end{pmatrix} \tag{3.46}$$

Remarque 5 *L'inégalité matricielle* (3.42) *est une réécriture du lemme 3 appliquée à la boucle fermée de la figure 3.6. Comme indiqué dans la remarque précédente, cette expression est initialement non linéaire en les inconnues. Une fois la matrice X_{cl} connue, cette expression devient linéaire en les inconnus restantes, c'est à dire les matrices A_{k_i}, B_{k_i}, C_{k_i} et D_{k_i}. Il suffit alors de résoudre l'expression* (3.42) *pour reconstruire les correcteurs.*

Fonction de Lyapunov dépendante de θ

Nous présentons maintenant un second théorème de synthèse basé sur le lemme 4.

Théorème 2 (Condition de faisabilité)

D'après (Oliveira et al., 2002), sous les hypothèses 1, il existe un correcteur de la forme (3.34)*, avec $\tilde{x} \in \mathbb{R}^{n_s}$, qui garantit :*

– la stabilité interne de la boucle fermée,

– un gain $\mathscr{L}_2$-induit entre w et z borné par γ, i.e. $\|z\|_2 < \gamma\|w\|_2$.

s'il existe $\gamma \in \mathbb{R}$, des matrices X, L, Y, F, Q, R, S, J et deux matrices symétriques $P, H \in R^{n_s \times n_s}$ satisfaisant les N LMIs suivantes :

$$\begin{pmatrix} P_i & J & A_iX + B_uL & A_i + B_uRC_y & B_{wi} + B_uRD_{yw} & \mathbf{0} \\ \star & H & Q & YA_i + FC_y & YB_{wi} + FD_{yw} & \mathbf{0} \\ \star & \star & X + X^T - P_i & \boldsymbol{I} + S^T - J & \mathbf{0} & X^TC_{zi}^T + L^T + D_{zu}^T \\ \star & \star & \star & Y + Y^T - H & \mathbf{0} & C_{zi}^T + C_y^TR^TD_{zu}^T \\ \star & \star & \star & \star & \boldsymbol{I} & D_{zwi}^T + D_{yw}^TR^TD_{zu}^T \\ \star & \star & \star & \star & \star & \gamma^2\boldsymbol{I} \end{pmatrix}$$

$$< 0, \quad i = 1 \dots N \tag{3.47}$$

$$\tag{3.48}$$

Remarque 6 *Comme dans le cas précédent, l'application du lemme d'analyse sur la boucle fermée de la figure 3.6 conduit à une inégalité matricielle non linéaire en les inconnues. Contrairement au théorème 1 la linéarisation du problème passe par un changement de variables assez complexe. L'avantage est que la reconstruction des matrices du correcteur est simplifiée en appliquant le changement de variables inverse.*

Reconstruction du correcteur A partir des matrices X et Y on construit les matrices U et V par une décomposition en valeurs singulières :

$$VU = I - YX = N\Sigma M^T = (N\Sigma^{\frac{1}{2}})(M\Sigma^{\frac{1}{2}})^T \tag{3.49}$$

Puis chaque correcteur sommet est reconstruit par l'expression suivante :

$$\begin{pmatrix} A_{k_i} & B_{k_i} \\ C_{k_i} & D_{k_i} \end{pmatrix} = \begin{pmatrix} V^{-1} & -V^{-1}YB_u \\ 0 & I \end{pmatrix} \begin{pmatrix} Q - YA_iX & F \\ L & R \end{pmatrix} \begin{pmatrix} U^{-1} & 0 \\ -C_yXU^{-1} & I \end{pmatrix} \tag{3.50}$$

Conclusion du chapitre

Ce chapitre a permis dans un premier temps d'introduire la commande H_∞ pour les systèmes LTI et notamment la quantification des performances et la formulation des objectifs de synthèse.

Dans un deuxième temps nous avons présenté deux théorèmes de synthèse de correcteurs pour les systèmes LPV à temps discret sous forme polytopique. Ces synthèses sont appliquées, dans le chapitre suivant, à la réalisation d'un correcteur à période variable.

Chapitre 4

Synthèse de correcteurs à période variable

Dans ce chapitre nous proposons une méthodologie pour obtenir un correcteur à période d'échantillonnage variable. Après le rappel des objectifs, l'approche est détaillée puis illustrée sur un exemple académique. Dans un dernier paragraphe nous proposons une autre formulation, actuellement inachevée, pour résoudre la même problématique.

4.1 Objectifs

Le premier chapitre a mis en évidence l'intérêt d'une loi de commande dont la période peut être ajustée au cours du temps. Le chapitre précédent a introduit les systèmes linéaires à paramètres variants et la synthèse de correcteurs pouvant s'adapter à la variation de paramètres mesurables. Nous proposons dans ce chapitre d'appliquer les outils de modélisation et de contrôle des systèmes LPV à la synthèse d'un correcteur dont la période d'échantillonnage peut varier.

Parmi les travaux sur la problématique des correcteurs à période variable, Cervin (2003) illustre sur un exemple l'intérêt de recalculer les paramètres d'un correcteur PID à temps discret face à une gigue sur la période d'échantillonnage. Une approche similaire est présentée dans (Albertos and Crespo, 1999). Dans (Albertos et al., 2003) les auteurs étudient la mise à jour de l'état interne du contrôleur lors de la permutation entre deux contrôleurs de période différente. L'objectif est de limiter la perturbation introduite par cette transition. L'approche utilise une interpolation et une optimisation afin reconstruire l'état pour la nouvelle période d'échantillonnage. Sala (2005) propose un correcteur à période variable basé sur la combinaison d'un retour d'état et d'un observateur tous deux dépendants de la période. La synthèse de leur gain respectif passe par la résolution d'un problème de faisabilité sous contrainte LMI. Néanmoins ce dernier étant de dimension infinie les auteurs utilisent

une technique de "maillage" pour le résoudre. Citons aussi le résultat d'analyse de stabilité de (Hu and Michel, 2000) applicable à l'interconnexion d'un système non-linéaire et d'un contrôleur non-linéaire via un échantillonneur et un bloqueur d'ordre zéro.

Afin d'obtenir un correcteur à période d'échantillonnage variable, nous avons tout d'abord étudié le problème dans une approche polynomiale de la synthèse par placement de pôles. Ce travail est présenté dans (Robert et al., 2005) où un correcteur de type RST à période variable est obtenu par la résolution formelle de la synthèse traditionnellement utilisée. La solution présentée se limite aux systèmes du deuxième ordre. Nous ne détaillerons pas cette approche, mais le lecteur peut se reporter à la publication disponible en annexe.

La méthode présentée ici repose sur la modélisation, sous forme d'un système LPV à temps discret, de la discrétisation paramétrée d'un modèle linéaire à temps continu. A partir de ce modèle, où la période est le paramètre variant, un correcteur à période variable est obtenu par la synthèse d'un correcteur LPV.

L'idée vient du constat qu'une solution pour avoir un correcteur à période variable est de synthétiser plusieurs correcteurs, chacun pour une période différente, et de choisir, à l'exécution, le bon correcteur en fonction de la valeur de la période. Cependant cette solution ne garantie ni la stabilité, ni la continuité de la commande aux changements de correcteur. De plus l'évolution de la performance peut être difficile à prévoir. Enfin le nombre de correcteurs augmente rapidement en fonction de l'intervalle des périodes admissibles et de la granularité recherchée, ce qui complique la synthèse et l'implémentation.

On reconnaît ici les problèmes des commandes à gains séquencés qui ont conduit à l'étude des systèmes LPV (voir chapitre 3). Notre idée est alors d'appliquer ce cadre théorique et méthodologique à notre cas.

4.2 Discrétisaion paramétrée d'un modèle continu

Cette section est consacrée à l'obtention d'un modèle LPV à temps discret pour représenter la discrétisation paramétrée d'un modèle LTI à temps continu, sur un intervalle de périodes d'échantillonnage connu et borné.

4.2.1 Rappels sur la discrétisation

Soit un système LTI G à temps continu, défini par la représentation d'état (4.1) avec $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ et $y \in \mathbb{R}^p$.

$$G : \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{4.1}$$

La représentation d'état du système G_d à temps discret, correspondant à la discrétisation exacte du système G précédé d'un bloqueur d'ordre zéro, est définie par l'équation (4.2) avec h la période d'échantillonnage et $x_k \in \mathbb{R}^n$ (Åström and Wittenmark, 1997).

$$G_d : \begin{cases} x_{k+1} = A_d x_k + B_d u_k \\ y_k = C_d x_k + D_d u_k \end{cases} \tag{4.2}$$

avec

$$\begin{aligned} A_d &= e^{Ah} \quad B_d = \int_0^h e^{A\tau} d\tau B \\ C_d &= C \qquad D_d = D \end{aligned} \tag{4.3}$$

Si la matrice A est inversible, B_d est aussi définie par :

$$B_d = A^{-1}(A_d - I)B \tag{4.4}$$

La méthode numérique usuelle pour obtenir les matrices A_d et B_d utilise l'exponentielle de la matrice M de l'équation (4.5). La matrice N contient alors A_d et B_d. L'avantage de cette méthode est de ne pas utiliser l'inversion de la matrice A.

$$\text{Soit } M = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} \qquad \text{alors} \qquad N = e^{Mh} = \begin{bmatrix} A_d & B_d \\ 0 & I \end{bmatrix} \tag{4.5}$$

Commandabilité et observabilité Un soin particulier doit être apporté dans le choix de la période d'échantillonnage. En effet, pour certaines valeurs dites "pathologiques", le système à temps discret n'est plus commandable ou observable.

D'après (Chen and Francis, 1995) la pulsation d'échantillonnage ω_s est pathologique si la matrice d'état A possède deux valeurs propres de partie réelle égale et de partie complexe qui diffère d'un multiple entier de la pulsation d'échantillonnage. Dans le cas contraire, la pulsation d'échantillonnage est dite non pathologique.

Exemple (Chen and Francis, 1995) Supposons que la matrice A ait pour valeurs propres : 0, 0, $\pm j$, $1 \pm 2j$. Elles sont regroupées sur deux lignes verticales d'abscisse $\mathrm{Re}(s) = 0$ et $\mathrm{Re}(s) = 1$. Sur la première ligne, la distance entre deux valeurs propres est 1 ou 2. Sur la seconde, cette distance est de 4. Pour ne pas être pathologique, la pulsation d'échantillonnage ne doit vérifier aucune des égalités $k\omega_s = 1$, $k\omega_s = 2$, $k\omega_s = 4$, $k \in \mathbb{N}^*$. Ainsi l'ensemble des pulsations d'échantillonnage interdites est défini par :

$$\left\{ \frac{1}{k} : k \geq 1 \right\} \cup \left\{ \frac{2}{k} : k \geq 1 \right\} \cup \left\{ \frac{4}{k} : k \geq 1 \right\} \tag{4.6}$$

Sachant que 4 est divisible par 2 et 1, cet ensemble est plus simplement défini par :

$$\left\{ \frac{4}{k} : k \geq 1 \right\} \tag{4.7}$$

Enfin notons que 4 est une borne supérieure de cette ensemble, ainsi la contrôlabilité et l'observabilité seront maintenues si la pulsation d'échantillonnage est supérieure à 4 rad/s.

Choix empirique de la période d'échantillonnage Le choix de la période d'échantillonnage est souvent basé sur des critères empiriques. Parmi ceux-ci citons l'ordre de grandeur basé sur le temps de montée t_m du système que l'on désire discrétiser (Åström and Wittenmark, 1997) :

$$\frac{t_m}{h} \approx 4 \ \ldots 10 \tag{4.8}$$

Pour un système du second ordre, d'amortissement $\xi = 0.7$ et de pulsation ω_0, cela équivaut à l'intervalle suivant :

$$\omega_0 h \approx 0.2 \ldots 0.6 \tag{4.9}$$

Remarque 7 *Pour ce système du second ordre, la période limite d'échantillonnage, selon le théorème de Nyquist-Shannon, est définie par* $\omega_0 h = \pi$. *L'expression* (4.9) *respecte donc très largement cette contrainte .*

4.2.2 Modèle affine fonction de la période d'échantillonnage

Nous avons présenté dans la section précédente la formule usuelle de discrétisation. Normalement la discrétisation se calcule pour une valeur donnée de la période h. Déterminer la représentation d'état à temps discret repose alors sur le calcul de la matrice N c'est à dire le calcul de l'exponentielle d'une matrice.

Dans notre cas h est une variable appartenant à l'intervalle $[h_{min}; h_{max}]$. L'objectif est d'obtenir un modèle à temps discret paramétré par la période h. Ce modèle doit être utilisable avec les outils de la commande LPV présentés au chapitre précédent.

La dépendance de A_d et B_d en h se fait ici au travers de l'exponentielle d'une matrice multipliée par h. Nous avons vu qu'un modèle LPV polytopique peut être obtenu à partir d'un modèle LPV affine en un vecteur de paramètres variants. L'objectif est alors d'exprimer $A_d(h)$ et $B_d(h)$ sous une forme affine en des paramètres variants.

Pour cela on propose d'approcher l'exponentielle e^{Ah} par une série de Taylor d'ordre M, on obtient :

$$A_d(h) = e^{Ah} \approx I + \sum_{i=1}^{M} \frac{A^i}{i!}\ h^i \tag{4.10}$$

$$B_d(h) = A^{-1}(A_d - I)B \approx \sum_{i=1}^{M} \frac{A^{i-1}B}{i!}\ h^i \tag{4.11}$$

Remarquons que ces deux expressions sont affines non par directement en h mais en ces puissances successives.

Localisation de l'approximation

L'approximation d'une fonction par une série de Taylor est valable localement. Les expressions (4.10) et (4.11) sont donc valables pour h proche de zéro. Dans notre cas, h est bornée par h_{min} et h_{max} avec $h_{min} > 0$ puisqu'une période de valeur 0 est irréaliste. Il est donc préférable de localiser l'approximation pour une valeur comprise entre h_{min} et h_{max}. En décomposant la période h comme suit :

$$h = h_0 + \delta \qquad \text{avec} \qquad h_{min} - h_0 \leq \delta \leq h_{max} - h_0 \tag{4.12}$$

De l'équation (4.5) on a :

$$\begin{pmatrix} A_d & B_d \\ 0 & I \end{pmatrix} = e^{Mh} = e^{Mh_0}e^{M\delta} = \begin{pmatrix} A_{h_0} & B_{h_0} \\ 0 & I \end{pmatrix} \begin{pmatrix} A_\delta & B_\delta \\ 0 & I \end{pmatrix} \tag{4.13}$$

On en déduit :

$$\begin{aligned} A_d &= A_{h_0}A_\delta \\ B_d &= B_{h_0} + A_{h_0}B_\delta \end{aligned} \tag{4.14}$$

Remarque 8 *Si $\delta = 0$, alors $A_\delta = I$ et $B_\delta = 0$ et on retrouve bien $A_d = A_{h_0}$ et $B_d = B_{h_0}$.*

Sachant que h_0 est connue à la conception, A_{h_0} et B_{h_0} sont des constantes. En appliquant la décomposition en série de Taylor sur la partie variable, c'est à dire A_δ et B_δ, les expressions (4.10) et (4.11) sont approchées par :

$$A_d(h) \approx A_{h_0}(I + \sum_{i=1}^{M} \frac{A^i}{i!}\ \delta^i) := A_d(\delta) \tag{4.15}$$

$$B_d(h) \approx B_{h_0} + A_{h_0}(\sum_{i=1}^{M} \frac{A^{i-1}B}{i!}\ \delta^i) := B_d(\delta) \tag{4.16}$$

Erreur d'approximation

L'usage d'une série de Taylor induit une erreur d'approximation. Pour quantifier cette dernière, en fonction de l'ordre de la série, on propose le critère de l'équation (4.17). Il évalue l'écart, au sens de la norme H_∞, entre $G_d(z)$ et $\tilde{G}_{d_M}(z)$, les discrétisations respectivement exacte et approchée d'un modèle continu $G(p)$, pour une valeur fixe de la période d'échantillonnage. La représentation d'état de $G_d(z)$ est obtenue par l'expression (4.3), celle de $\tilde{G}_{d_M}(z)$ par (4.15) et (4.16).

$$J_M(h) = \|G_d(z,h) - \tilde{G}_{d_M}(z,h)\|_\infty \tag{4.17}$$

La différence entre les deux discrétisations provient uniquement du calcul de l'exponentielle e^{Ah} qui est soit exact soit approché par une série de Taylor. Nous pourrions nous contenter de quantifier, selon un critère à définir, l'erreur numérique entre le calcul exact et approché de cette exponentielle. Néanmoins, notre objectif étant de concevoir une loi de commande, il est plus pertinent d'utiliser l'erreur entre le modèle approché, utilisé pour la synthèse, et le modèle exact, inutilisable dans l'approche que nous proposons.

Notons enfin qu'à partir du critère (4.17) on peut modéliser l'erreur d'approximation par une incertitude non paramétrique et étudier a posteriori la robustesse du contrôleur vis à vis de cette erreur d'approximation.

L'évaluation du critère (4.17) pour plusieurs valeurs de la période d'échantillonnage et de l'ordre de la série permet de choisir empiriquement la valeur de M pour que l'erreur soit inférieure à un seuil que l'on se fixe.

Exemple On illustre ici l'effet de l'approximation, au sens du critère (4.17), sur la discrétisation d'un système du deuxième ordre pour une période h comprise entre 40 et 120 ms. Les deux graphiques 4.1 et 4.2 se distinguent par la valeur de h_0, c'est à dire la localisation du développement en série de Taylor.

Comme on peut s'y attendre, l'erreur d'approximation est d'autant plus grande qu'on s'éloigne de la valeur h_0. De même, plus l'ordre M est élevé et plus l'erreur est faible. Notons que l'erreur maximale est supérieure quand $h_0 = h_{min}$. Rien d'illogique dans cette observation mais pour une valeur donnée de l'erreur, le choix de h_0 influence l'ordre de la série de Taylor.

4.2.3 Modèle polytopique fonction de la période d'échantillonnage

Les outils de synthèse, du chapitre 3, nécessitent un modèle sous forme polytopique. Nous présentons ici deux approches pour transformer un modèle affine en un modèle polytopique.

Dans un premier temps nous appliquons la méthodologie générale du chapitre 3. Notre modèle affine a la particularité d'avoir un vecteur de paramètres variants $H = [\delta, \delta^2, \ldots, \delta^M]$

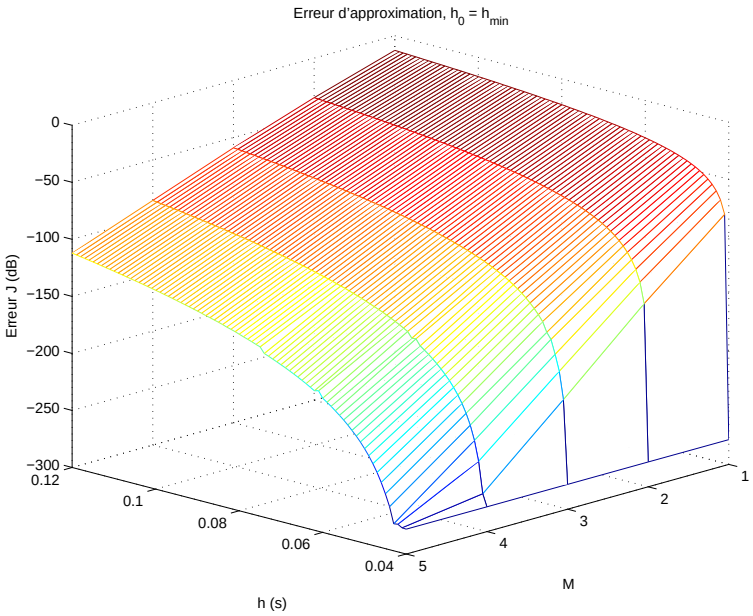


FIG. 4.1 – Erreur d'approximation, $h_0 = h_{min}$

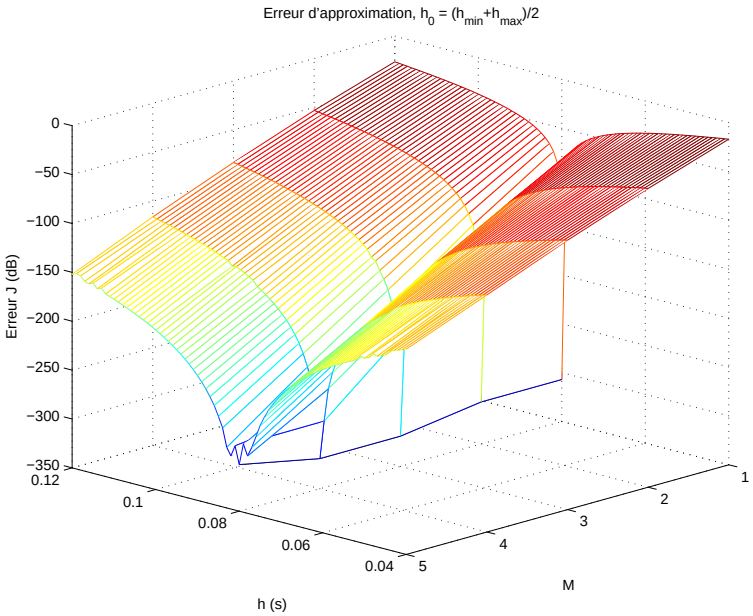


FIG. 4.2 – Erreur d'approximation, $h_0 = (h_{min} + h_{max})/2$

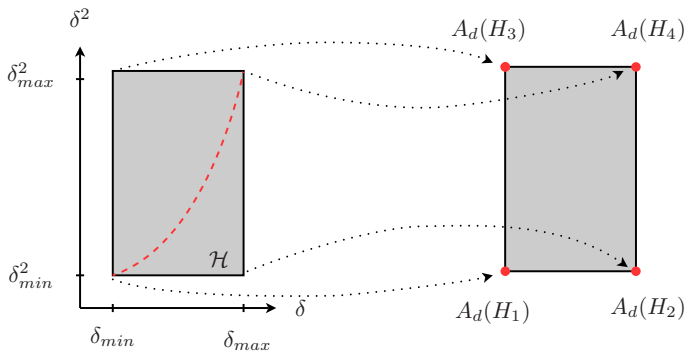


FIG. 4.3 – Exemple d'un polytope contenant $A_d(\delta)$ avec $\delta_{min} = 0$

dont les coordonnées sont dépendantes. Dans une seconde approche nous utilisons cette dépendance pour réduire le nombre de sommets du modèle polytopique.

Modèle polytopique simple

On applique ici la méthode du paragraphe 3.2.1.

Sachant que la période est bornée par h_{min} et h_{max}, le vecteur $H = [\delta, \delta^2, \ldots, \delta^M]$ appartient à un hypercube $\mathcal{H}$ de 2^M sommets. Chaque sommet est défini par un vecteur $[\nu_1, \nu_2, \ldots, \nu_M]$ où l'élément ν_i peut prendre l'une des deux valeurs δ^i_{min} ou δ^i_{max} avec $\delta_{min} = h_{min} - h_0$ et $\delta_{max} = h_{max} - h_0$.

Les matrices $A_d(\delta)$ et $B_d(\delta)$ étant affines en le vecteur H, les matrices sommets A_{d_i} et B_{d_i} des polytopes contenant $A_d(\delta)$ et $B_d(\delta)$ sont obtenues par l'évaluation de $A_d(\delta)$ et $B_d(\delta)$ à chaque sommet de l'hypercube $\mathcal{H}$. La figure 4.3 illustre le principe pour $M = 2$ avec

$$
\begin{aligned}
A_{d_1} &= A_d(H_1) \quad \text{où} \quad H_1 = [\delta_{min}, \delta^2_{min}] \\
A_{d_2} &= A_d(H_2) \quad \text{où} \quad H_2 = [\delta_{max}, \delta^2_{min}] \\
A_{d_3} &= A_d(H_3) \quad \text{où} \quad H_3 = [\delta_{min}, \delta^2_{max}] \\
A_{d_4} &= A_d(H_4) \quad \text{où} \quad H_4 = [\delta_{max}, \delta^2_{max}] \\
\mathcal{H} &= \mathbf{Co}\{H_1, H_2, H_3, H_4\}
\end{aligned}
$$

Cette approche ne tient pas compte de la dépendance entre $\delta, \delta^2, \ldots, \delta^M$. Sur l'exemple de la figure 4.3, l'ensemble des points $\{[\delta, \delta^2], 0 \leq \delta \leq \delta_{max}\}$, représenté par la courbe rouge en pointillés, est très grossièrement approché par le rectangle gris. Le modèle discret à période variable est alors contenu dans un ensemble de systèmes beaucoup plus grand. La synthèse du correcteur étant contrainte par cet ensemble "trop grand", cette modélisation introduit forcément un conservatisme.

De plus le nombre de sommets étant 2^M, la dimension du polytope augmente très rapidement. Or d'une part les théorèmes (1) et (2) nécessitent la faisabilité simultanée de respectivement $2 * 2^M + 1$ et 2^M LMIs. D'autre part le correcteur à implémenter s'exprime comme la combinaison convexe de 2^M correcteurs sommets. La complexité de la synthèse et de la reconstruction du correcteur en ligne sont donc liées à la dimension du polytope. Pour cette raison il est intéressant de la réduire.

Modèle polytopique réduit

On propose de réduire le nombre de sommets du polytope $\mathcal{H}$ en utilisant la dépendance entre δ, δ^2, ... et δ^N. Cette réduction n'est valable que pour $\delta_{min} = 0$.

Sur l'exemple de la figure 4.3, la parabole est hors du triangle formé par les sommets $\{0, 0\}$, $\{0, \delta^2_{max}\}$ et $\{\delta_{max}, \delta^2_{max}\}$. On peut donc supprimer le sommet $\{0, \delta^2_{max}\}$.

Cette idée peut être étendue à un polytope de dimension N. En posant :

$$h = h_{min} + \delta, \quad 0 \leq \delta \leq \delta_{max}, \quad \delta_{max} = h_{max} - h_{min}, \tag{4.18}$$

l'inégalité suivante est toujours vérifiée :

$$\delta \ \delta^n \leq \frac{\delta^{n+1}_{max}}{\delta^n_{max}} \ \delta^n \quad \text{i.e.} \quad \delta^{n+1} \leq \delta_{max} \ \delta^n \tag{4.19}$$

Les sommets dont les coordonnées ne vérifient pas cette expression peuvent être supprimés. Pour cela, sachant que les sommets H_i de $\mathcal{H}$ sont définis par un vecteur $(\nu_1, \nu_2, \ldots, \nu_M)$ où $\nu_i = 0$ ou δ^i_{max} suivant le sommet considéré, il suffit de vérifier l'inégalité suivante :

$$\nu_{n+1} \leq \delta_{max} \ \nu_n \tag{4.20}$$

Les sommets admissibles ont la forme suivante :

$$\begin{aligned} &(0, 0, 0, \ldots, 0) \\ &(\delta_{max}, 0, 0, \ldots, 0) \\ &(\delta_{max}, \delta^2_{max}, 0, \ldots, 0) \\ &\vdots \\ &(\delta_{max}, \delta^2_{max}, \delta^3_{max}, \ldots, \delta^M_{max}) \end{aligned} \tag{4.21}$$

Remarque 9 *Le sommet de coordonnées* $(0, \delta^2_{max}, 0, \ldots, 0)$ *est l'exemple d'un sommet ne vérifiant pas l'inégalité* (4.20).

On obtient $M + 1$ sommets au lieu de 2^M. Notons que ces sommets sont linéairement indépendants et forment un simplexe, qui est par définition un polytope (Boyd and Vandenberghe, 2004).

Exemple Les figures 4.4 et 4.5 illustrent la réduction du polytope pour des séries de Taylor respectivement d'ordre 2 et 3. Dans le premier cas, le rectangle est réduit à un triangle. Dans le deuxième cas le cube est réduit à une pyramide.

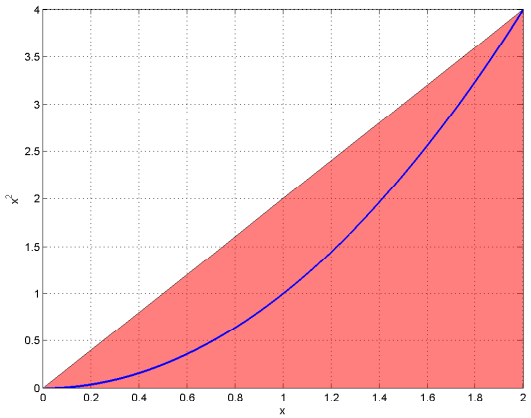


FIG. 4.4 – Réduction pour M=2

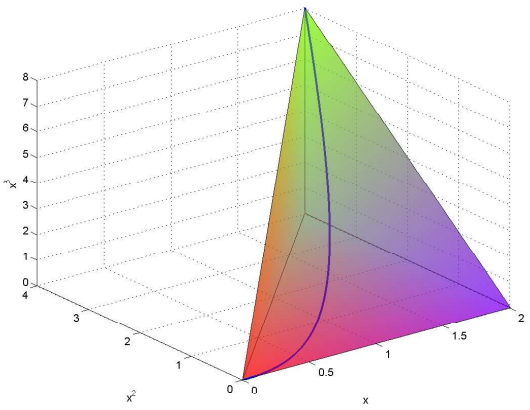


FIG. 4.5 – Réduction pour M=3

Reconstruction des coordonnées polytopiques

La calcul des coordonnées polytopiques est nécessaire pour l'évaluation du modèle LPV polytopique et pour la reconstruction du correcteur pour une valeur donnée de la période h.

Les coordonnées polytopiques sont obtenues par la résolution d'un système d'équations linéaires (voir le paragraphe 3.2.1). Dans le cas du modèle polytopique simple, le système est sous-dimensionné et on utilise l'algorithme du paragraphe 3.2.1. Pour un modèle polytopique réduit, la réduction du nombre de sommets rend le système d'équations inversible et on peut le résoudre avec la méthode de son choix.

4.3 Formulation de la synthèse

Pour spécifier les objectifs de la boucle fermée nous utilisons la méthodologie présentée au paragraphe 3.1.3. Le modèle LPV du procédé étant à temps discret il en sera de même pour le système augmenté. Avant de construire ce dernier il faut donc discrétiser les gabarits fréquentiels.

4.3.1 Discrétisation des gabarits fréquentiels

Dans notre cas les objectifs de la boucle fermée sont exprimés par des gabarits fréquentiels. Si la synthèse est à temps continu, ces gabarits sont directement utilisables. Dans le cas

d'une synthèse à temps discret, il est nécessaire de les discrétiser. Pour une période d'échantillonnage constante, cette étape est très simple. Dans notre cas, la période d'échantillonnage variable impose une discrétisation paramétrée des objectifs.

Le choix de la période d'échantillonnage repose traditionnellement sur des critères empiriques qui relient la performance en boucle fermée à la période (voir paragraphe 4.2.1). Si la période d'échantillonnage doit être adaptée à la performance en boucle fermée, alors réciproquement, la performance en boucle fermée doit être adaptée à la période d'échantillonnage quand cette dernière varie. On propose donc d'ajuster les gabarits fréquentiels W_I et W_O, qui définissent les objectifs en boucle fermée, à la période d'échantillonnage.

Remarque 10 *Dans un contexte où les ressources allouées à une loi de commande fluctuent, conduisant à la variation de la période, adapter l'objectif en boucle fermée permet de maîtriser la dégradation de la performance au profit d'une stabilité garantie.*

Méthodologie

Les gabarits $W_I(p)$ et $W_O(p)$ sont factorisés en deux parties :

- L'une contenant des pôles et des zéros constants. Cela permet par exemple de compenser des oscillations ou des modes souples qui, par définition, ont une pulsation indépendante de la période d'échantillonnage.
- L'autre contenant les pôles et les zéros permettant d'agir sur la bande passante du gabarit. Ce sont les plus rapides. Ils sont définis par une fonction linéaire en la fréquence d'échantillonnage pour pouvoir adapter la bande passante du gabarit en fonction de la période d'échantillonnage. Dans l'approche proposée, ces pôles et zéros sont réels.

La partie constante est combinée avec le modèle à temps continu du système pour donner un modèle augmenté à temps continu. Ce dernier est ensuite discrétisé avec l'une des méthodes exposées dans la section 4.2.

La partie variable, nommée $V(p)$, est discrétisée avec la méthode suivante :

1. Factoriser $V(p)$ comme le produit de systèmes du premier ordre. Les pôles et les zéros sont définis par une fonction linéaire en la fréquence $f = 1/h$, i.e. $z_i = b_i f$ et $p_i = a_i f$ avec $a_i, b_i \in \mathbb{R}$.
$$V(p) = \beta \prod_i \frac{p - b_i f}{p - a_i f} = \beta \prod_i V_i(p) \tag{4.22}$$
2. Mettre sous la forme canonique observable chaque $V_i(p)$:
$$V_i(p) : \begin{cases} \dot{x}_i = a_i f \, x_i + (a_i f - b_i f) \, u_i \\ y_i = x_i + u_i \end{cases} \tag{4.23}$$

3. Former l'interconnexion série des systèmes $V_i(p)$. Ceci permet d'obtenir $V(p)$ défini par l'expression (4.24) où les matrices A_v, B_v, C_v et D_v sont des matrices réelles de dimensions appropriées (voir remarque 12).

$$V(p) : \begin{cases} \dot{x}_v = A_v f\, x_v + B_v f\, \beta u_v \\ y_v = C_v x_v + D_v\, \beta u_v \end{cases} \tag{4.24}$$

4. Discrétiser la représentation d'état de l'expression (4.24). L'équation de l'état étant linéaire en f et celle de la sortie constante, la représentation d'état à temps discret $V_d(z)$ devient, après discrétisation exacte, indépendante de la période d'échantillonnage, comme l'illustre l'équation (4.26).

$$V_d(z) : \begin{cases} x_{k+1} = A_{v_d}\, x_k + B_{v_d}\, \beta u_k \\ y_k = C_v\, x_k + D_v\, \beta u_k \end{cases} \tag{4.25}$$

$$\begin{cases} A_{v_d} = e^{A_v f\, h} = e^{A_v} \\ B_{v_d} = (A_v f)^{-1}(A_{v_d} - I) B_v f = A_v^{-1}(A_{v_d} - I) B_v \end{cases} \tag{4.26}$$

Remarque 11 *L'utilisation, à l'étape 2, de la forme canonique observable permet à l'équation de sortie d'être indépendante de f et facilite ainsi la discrétisation à l'étape 4.*

Remarque 12 *L'interconnexion* ***série*** *de deux systèmes de la forme* (4.27) *conduit à un système de la même forme défini par* (4.28).

$$\begin{cases} \dot{x} = A f\, x + B f\, u \\ y = C\, x + D\, u \end{cases} \tag{4.27}$$

$$\begin{aligned} A &= \begin{pmatrix} A_1 & 0 \\ B_2 C_1 & A_2 \end{pmatrix} \quad B = \begin{pmatrix} B_1 \\ B_2 D_1 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ C &= \begin{pmatrix} D_2 C_1 & C_2 \end{pmatrix} \quad D = D_2 D_1 \end{aligned} \tag{4.28}$$

D'une part les matrices C et D s'expriment uniquement en fonction de C_i ou D_i, $i = 1, 2$, ce qui garantit une équation de sortie indépendante de f. D'autre part l'absence de produits entre A_i et B_i, $i = 1, 2$, maintient une dépendance de l'équation d'état linéaire en f. L'interconnexion série de 3 systèmes produit une représentation d'état (4.27) *avec les matrices suivantes :*

$$\begin{aligned} A &= \begin{pmatrix} A_1 & 0 & 0 \\ B_2 C_1 & A_2 & 0 \\ B_3 D_2 C_1 & B_3 C_2 & A_3 \end{pmatrix} \quad B = \begin{pmatrix} B_1 \\ B_2 D_1 \\ B_3 D_2 D_1 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \\ C &= \begin{pmatrix} D_3 D_2 C_1 & D_3 C_2 & C_3 \end{pmatrix} \quad D = D_3 D_2 D_1 \end{aligned} \tag{4.29}$$

Par construction, l'interconnexion ***série*** *de plus de trois systèmes de la forme* (4.27) *conduit à un système de la forme* (4.27). *Ainsi l'interconnexion des systèmes* $V_i(p)$, *qui ont une forme similaire à* (4.27), *produit le système* (4.24) *où la dépendance en* f *facilite la discrétisation de l'étape 4.*

La simplification entre f et h rend cette discrétisation particulièrement intéressante puisque le modèle discret des gabarits devient un modèle linéaire invariant dans le temps (LTI).

4.3.2 Construction du système augmenté

Nous venons de décrire la discrétisation du modèle du procédé et celle des gabarits fréquentiels. Il reste maintenant à interconnecter les deux pour construire le système augmenté de la figure 4.6. Pour plus de clarté, nous reprenons toutes les étapes avec une notation commune.

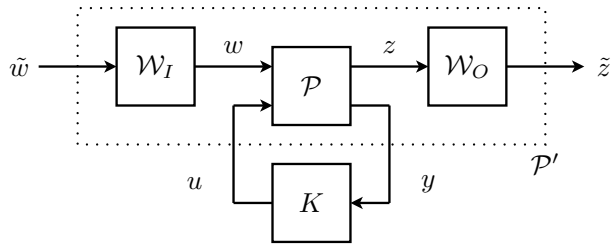


FIG. 4.6 – Système augmenté

Soit un système continu dont la représentation d'état est définie par l'expression (4.30) où $x \in R^n$ est le vecteur d'état, $w \in R^{m_w}$ est le vecteur des entrées exogènes, $u \in R^{m_u}$ est le vecteur des commandes, $z \in R^{p_z}$ est le vecteur des sorties exogènes et $y \in R^{p_y}$ est le vecteur des mesures.

$$P : \begin{cases} \dot{x}(t) = Ax(t) + B_w w(t) + B_u u(t) \\ z(t) = C_z x(t) + D_{zw} w(t) + D_{zu} u(t) \\ y(t) = C_y x(t) + D_{yw} w(t) + D_{yu} u(t) \end{cases} \tag{4.30}$$

Ce modèle est discrétisé selon l'approche présentée dans le paragraphe 4.2 en posant :

$$\begin{array}{llll} A = A & B = \begin{pmatrix} B_w & B_u \end{pmatrix} & C = \begin{pmatrix} C_z \\ C_y \end{pmatrix} & D = \begin{pmatrix} D_{zw} & D_{zu} \\ D_{yw} & D_{yu} \end{pmatrix} \\ A_d = \mathcal{A} & B_d = \begin{pmatrix} \mathcal{B}_w & \mathcal{B}_u \end{pmatrix} & C_d = C & D_d = D \end{array} \tag{4.31}$$

La discrétisation paramétrée, utilisant une série de Taylor d'ordre M, conduit à la construction d'un polytope $\mathcal{H}$. Ce dernier est composé de l sommets où l est égal à 2^M pour le polytope simple et $M+1$ pour le polytope réduit. Chacun des l sommets est décrit par un vecteur ω_i de la forme $(\delta_1, \delta_2, \ldots, \delta_l)$ où $\delta_i = \delta^i_{min}$ ou δ^i_{max} en fonction du sommet considéré et de l'application de la réduction du polytope.

Le modèle LPV polytopique à temps discret obtenu est décrit par la représentation d'état (4.33) où la dépendance temporelle de w, u, z, y et H est supprimée par clarté et avec :

$$\begin{gathered} H = \begin{pmatrix} \delta & \delta^2 & \ldots & \delta^M \end{pmatrix} \qquad H \in \mathcal{H} = \mathbf{Co}\{\omega_1, \ldots, \omega_l\} \\ H = \sum_{i=1}^{l} \alpha_i \omega_i \qquad \mathcal{A}(H) = \sum_{i=1}^{l} \alpha_i \mathcal{A}_i \qquad \sum_{i=1}^{l} \alpha_i = 1 \qquad \alpha_i \geq 0 \end{gathered} \tag{4.32}$$

$$\mathcal{P}(H) : \begin{cases} x_{k+1} = \mathcal{A}(H)x_k + \mathcal{B}_w(H)w + \mathcal{B}_u(H)u \\ z = C_z x_k + D_{zw} w + D_{zu} u \\ y = C_y x_k + D_{yw} w + D_{yu} u \end{cases} \tag{4.33}$$

La discrétisation des parties variables des gabarits $W_I(p)$ et $W_O(p)$, selon la méthodologie précédemment exposée, produit deux systèmes à temps discret stationnaires décrits par les expressions (4.34) et (4.35) où la dimension des vecteurs d'état x_I et x_O dépend des gabarits choisis. Notons que la partie constante des gabarits doit être préalablement incluse dans le système à temps continu P.

$$\mathcal{W}_I : \begin{cases} x_{I_{k+1}} = A_I x_{I_k} + B_I \tilde{w} \\ w = C_I x_{I_k} + D_I \tilde{w} \end{cases} \tag{4.34}$$

$$\mathcal{W}_O : \begin{cases} x_{O_{k+1}} = A_O x_{O_k} + B_O z \\ \tilde{z} = C_O x_{O_k} + D_O z \end{cases} \tag{4.35}$$

Le système augmenté $\mathcal{P}'(H)$ est obtenu par l'interconnexion de $\mathcal{P}(H)$, $\mathcal{W}_I$ et $\mathcal{W}_O$. On obtient un système LPV polytopique à temps discret, dont l'état est $x'_k = (x_k \; x_{I_k} \; x_{O_k})^T$, qui est décrit par :

$$\mathcal{P}'(H) : \begin{cases} x'_{k+1} = \mathcal{A}'(H)x'_k + \mathcal{B}'_w(H)\tilde{w} + \mathcal{B}'_u(H)u \\ \tilde{z} = \mathcal{C}'_z x'_k + \mathcal{D}'_{zw}\tilde{w} + \mathcal{D}'_{zu} u \\ y = \mathcal{C}'_y x'_k + \mathcal{D}'_{yw}\tilde{w} + \mathcal{D}'_{yu} u \end{cases} \tag{4.36}$$

avec :

$$\mathcal{A}'(H) = \begin{pmatrix} \mathcal{A}(H) & \mathcal{B}_w(H)C_I & 0 \\ 0 & A_I & 0 \\ B_O C_z & B_O D_{zw} C_I & A_O \end{pmatrix} \quad \mathcal{B}'_w(H) = \begin{pmatrix} \mathcal{B}_w(H) D_I \\ B_I \\ B_O D_{zw} D_I \end{pmatrix} \quad \mathcal{B}'_u(H) = \begin{pmatrix} \mathcal{B}_u(H) \\ 0 \\ B_O D_{zu} \end{pmatrix}$$

$$\mathcal{C}'_z = \begin{pmatrix} D_O C_z & D_O D_{zw} C_I & C_O \end{pmatrix} \qquad \mathcal{D}'_{zw} = \begin{pmatrix} D_O D_{zw} D_I \end{pmatrix} \qquad \mathcal{D}'_{zu} = \begin{pmatrix} D_O D_{zu} \end{pmatrix}$$

$$\mathcal{C}'_y = \begin{pmatrix} C_y & D_{yw} C_I & 0 \end{pmatrix} \qquad \mathcal{D}'_{yw} = \begin{pmatrix} D_{yw} D_I \end{pmatrix} \qquad \mathcal{D}'_{yu} = \begin{pmatrix} D_{yu} \end{pmatrix}$$

Les outils de synthèse, présentés au chapitre précédent, sont alors appliqués sur la représentation d'état (4.36), après vérification des hypothèses 1 :

- H_1 : $\mathcal{D}'_{yu} = 0$.
- H_2 : $\mathcal{B}'_u(H)$, $\mathcal{C}'_y$, $\mathcal{D}'_{zu}$ et $\mathcal{D}'_{yw}$ indépendants de H.
- H_3 : Le système $\mathcal{P}'(H)$ doit être quadratiquement stabilisable et détectable quelle que soit la trajectoire de H. Cette hypothèse peut se vérifier avec le test des inégalités matricielles (3.35) et (3.36).

Remarque 13 *Dans le cas où $\mathcal{B}_u$ n'est pas constant, il est possible de modifier le système pour satisfaire l'hypothèse H_2. Pour cela, Apkarian et al. (1995) proposent d'ajouter, en entrée du système, un filtre à temps discret du premier ordre strictement propre. Si le système possède plusieurs commandes, on applique un filtre par commande. L'ensemble des filtres est modélisé par le système* (4.37) *où les matrices ont des dimensions appropriées.*

$$\mathcal{F} : \begin{cases} x_{f_{k+1}} = A_f x_{f_k} + B_f u \\ y_f = C_f x_{f_k} \end{cases} \tag{4.37}$$

Après l'interconnexion avec $\mathcal{F}$, les matrices du système augmenté $\mathcal{P}'(H)$ deviennent :

$$\mathcal{A}'(H) = \begin{pmatrix} \mathcal{A}(H) & \mathcal{B}_w(H)C_I & 0 & \mathcal{B}_u(H)C_f \\ 0 & A_I & 0 & 0 \\ B_O C_z & B_O D_{zw} C_I & A_O & B_O D_{zu} C_f \\ 0 & 0 & 0 & A_f \end{pmatrix} \quad \mathcal{B}'_w(H) = \begin{pmatrix} \mathcal{B}_w(H) D_I \\ B_I \\ B_O D_{zw} D_I \\ 0 \end{pmatrix} \quad \mathcal{B}'_u(H) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ B_f \end{pmatrix}$$

$$\mathcal{C}'_z = \begin{pmatrix} D_O C_z & D_O D_{zw} C_I & C_O & D_O D_{zu} C_f \end{pmatrix} \qquad \mathcal{D}'_{zw} = \begin{pmatrix} D_O D_{zw} D_I \end{pmatrix} \qquad \mathcal{D}'_{zu} = \begin{pmatrix} 0 \end{pmatrix}$$

$$\mathcal{C}'_y = \begin{pmatrix} C_y & D_{yw} C_I & 0 & D_{yu} C_f \end{pmatrix} \qquad \mathcal{D}'_{yw} = \begin{pmatrix} D_{yw} D_I \end{pmatrix} \qquad \mathcal{D}'_{yu} = \begin{pmatrix} 0 \end{pmatrix}$$

Sachant que la matrice B_f est indépendante de H l'hypothèse H_2 est alors vérifiée, ceci au détriment de l'augmentation de l'ordre du système augmenté.

La bande passante des filtres doit être négligeable devant la dynamique du procédé. Pour obtenir la plus haute bande passante on peut placer le pôle du filtre à zéro ce qui revient à un retard variable d'une période d'échantillonnage.

4.4 Exemple d'illustration

Nous illustrons maintenant notre approche sur un exemple académique. Le procédé est un pendule stable du deuxième ordre dont la linéarisation autour du point d'équilibre produit la représentation d'état suivante :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \qquad x = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} \tag{4.38}$$

où

$$\begin{aligned} A &= \begin{pmatrix} 0 & 1 \\ \frac{-g}{l} & \frac{-fv}{m\ l^2} \end{pmatrix} \quad B = \begin{pmatrix} 0 \\ \frac{1}{m\ l^2} \end{pmatrix} \\ C &= \begin{pmatrix} 1 & 0 \end{pmatrix} \end{aligned} \tag{4.39}$$

avec θ l'angle du pendule, l sa longueur, m sa masse, f_v un coefficient de frottement visqueux et g l'accélération gravitationnelle. Avec $l = 1$, $m = 1$ et $f_v = 3$, le système possède une paire de pôles complexes conjugués de pulsation $w_0 = 3,13$ rad/s et d'amortissement $\xi = 0,48$. La réponse fréquentielle du modèle est présentée sur la figure 4.7.

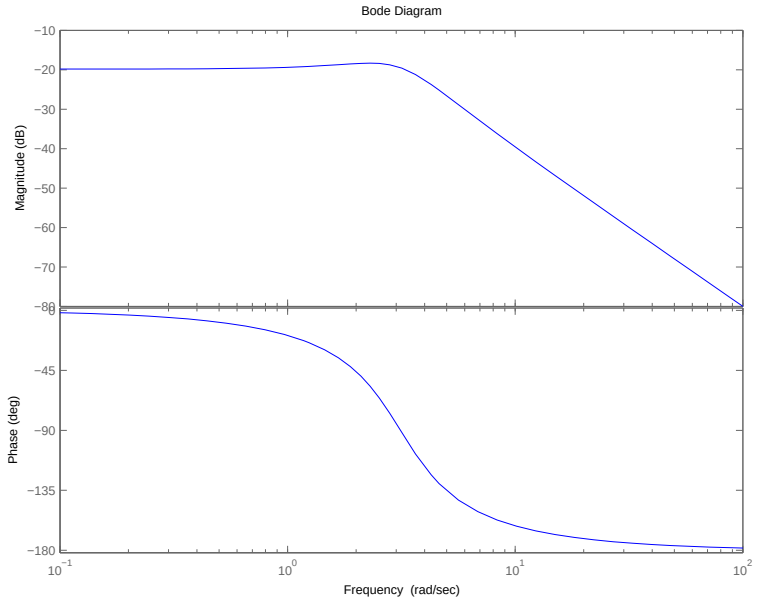


FIG. 4.7 – Réponse fréquentielle du modèle linéaire du pendule

Remarque 14 *L'étude de cet exemple est basée sur le modèle linéaire de l'équation* (4.38).

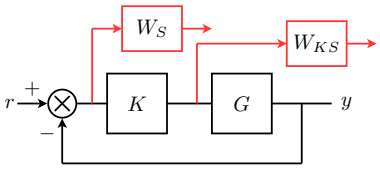
4.4.1 Objectifs en boucle fermée

L'objectif principal de la boucle fermée est d'augmenter l'amortissement du système pour éliminer les oscillations.

Nous nous plaçons dans un contexte où la période de la commande peut être ajustée en réponse à des variations d'allocation de ressources. L'idée est d'apporter une flexibilité en permettant d'ajuster la période de la commande, tout en maintenant la stabilité du procédé et si possible en respectant le cahier des charges de la boucle fermée. Nous choisissons de pouvoir augmenter la période dans un rapport de 3. Ainsi, si la commande utilise 100% des ressources d'un calculateur, on pourra en libérer jusqu'à 60% pour d'autres activités.

Dans un premier temps, nous concevons un correcteur à période constante en utilisant une synthèse H_∞ traditionnelle. Ce correcteur servira de référence. Dans un deuxième temps le correcteur à période variable est synthétisé selon la méthodologie précédemment exposée. Les objectifs de cette seconde synthèse, sont d'une part, d'obtenir un comportement nominal (pour la plus petite période) identique au correcteur de référence. D'autre part, sachant que la période peut être multipliée par 3, le comportement de la boucle fermée, pour la plus grande période, devra être dégradé dans la même proportion.

Dans le cadre de l'approche H_∞ que l'on considère, on choisit de résoudre le problème usuel de sensibilité mixte de l'expression (4.40) illustré sur la figure 4.8.



$$\left\| \begin{matrix} W_S(p)\, S(p) \\ W_{KS}(p)\, KS(p) \end{matrix} \right\|_\infty \leq \gamma \qquad (4.40)$$

FIG. 4.8 – Problème de sensibilité mixte étudié

$G(p)$ est la fonction de transfert du procédé, $K(p)$ celle du correcteur et les fonctions de sensibilité $S(p)$ et $KS(p)$ sont définies par :

$$S(p) = \frac{1}{1 + G(p)\, K(p)} \qquad KS(p) = \frac{K}{1 + G(p)\, K(p)} \qquad (4.41)$$

L'objectif nominal en boucle fermée est formulé par les gabarits fréquentiels usuels suivants (Skogestad and Postlethwaite, 1996) :

$$W_S(p) = \frac{p\, M_S + \omega_S}{p + \omega_S\, \epsilon_S} \qquad W_{KS}(p) = \frac{p + \omega_{KS}/M_{KS}}{\epsilon_{KS}\, p + \omega_{KS}} \qquad (4.42)$$

4.4.2 Synthèse du correcteur de référence à période constante

Nous résolvons tout d'abord le problème à temps continu dont le résultat nous servira de référence. Une synthèse H_∞ est appliquée au problème de sensibilité mixte de l'équation (4.40) avec les gabarits fréquentiels (4.42) où $\omega_S = 4$ rad/s pour accélérer le système en boucle fermée, $M_S = 2$ pour garantir une bonne marge de module et $\epsilon_S = 0,005$ pour borner l'erreur statique. $\omega_{KS} = 50$ rad/s, $M_{KS} = 20$ et $\epsilon_{KS} = 0,01$ autorisent un fort gain pour la commande en basses fréquences et atténuent les bruits sur la commande en hautes fréquences.

A l'issue de la synthèse, $\gamma = 1,18$ et les fonctions de sensibilité obtenues sont présentées sur la figure 4.9.

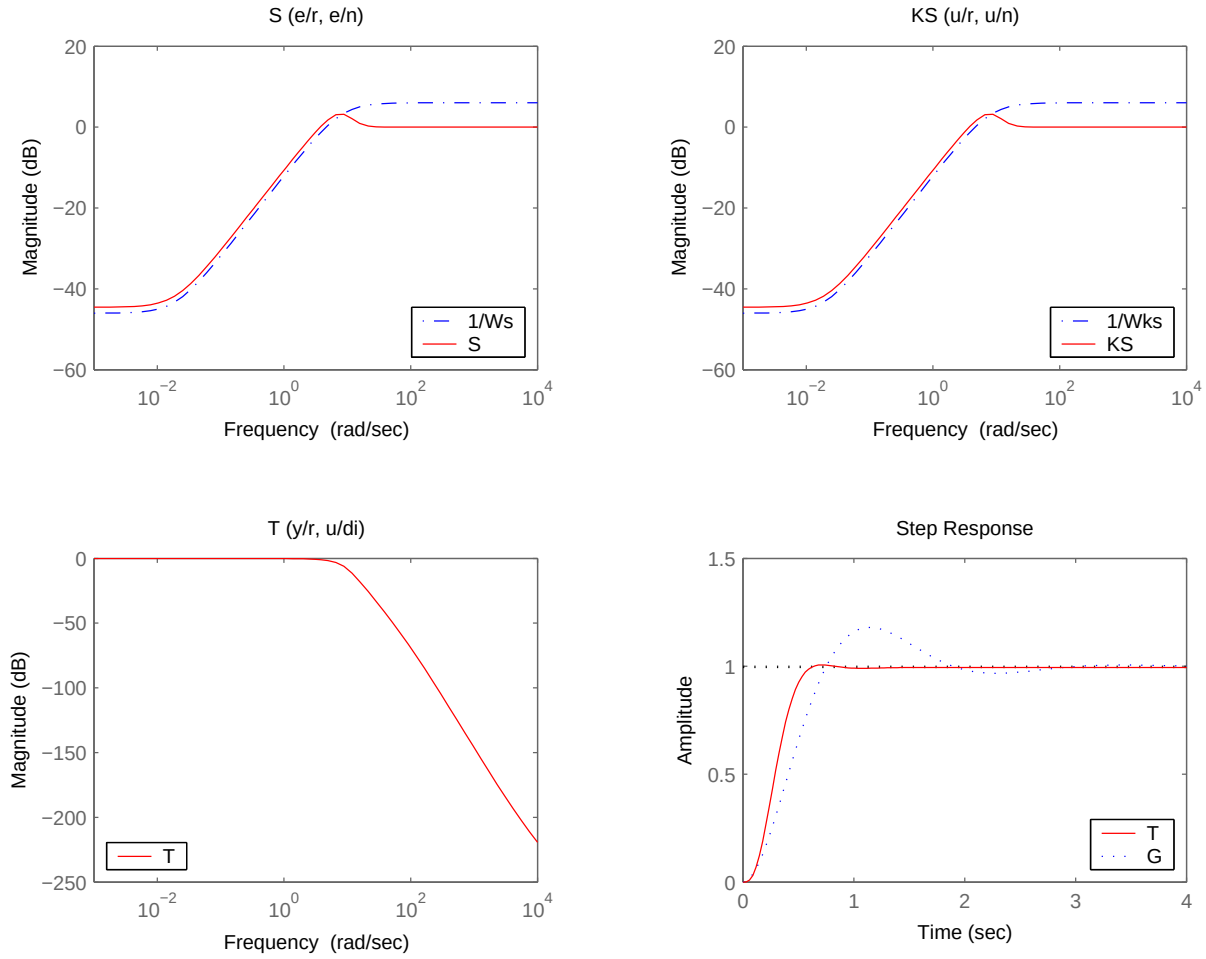


FIG. 4.9 – Sensibilités et réponse indicielle à l'issue de la synthèse de référence

On observe sur la sensibilité S une erreur statique inférieure à 0,006 et un pic de sensibilité

inférieur à 1,4 ce qui induit une bonne marge de robustesse. Les marges sont de 66° pour la phase et 13 dB pour le gain. Sans des connaissances approfondies sur la référence et les bruits que pourrait subir le procédé il est difficile de commenter la sensibilité KS. Néanmoins elle respecte un profil classique avec un maximum inférieur à 11,4 dB et une bande passante de 63 rad/s. Sachant que la bande passante du procédé en boucle ouverte est de 3,13 rad/s, qu'on cherche à accélérer le procédé, et que la bande passante de KS est supérieure d'environ une décade, ces valeurs semblent cohérentes. La boucle fermée (T) a une bande passante de 6,5 rad/s. Bien que le tracé de T ne montre pas de dépassements, l'étude des pôles et des zéros indique la présence d'une paire de pôles complexes conjugués et d'un zéro instable, ce qui se traduit par de faibles oscillations sur la réponse indicielle. Enfin le temps de réponse, à 5 %, est passé de 1,68 s en boucle ouverte à 0,52 s en boucle fermée et le dépassement a été réduit à sensiblement 1%. Ces résultats remplissent honorablement l'objectif que nous nous sommes fixé.

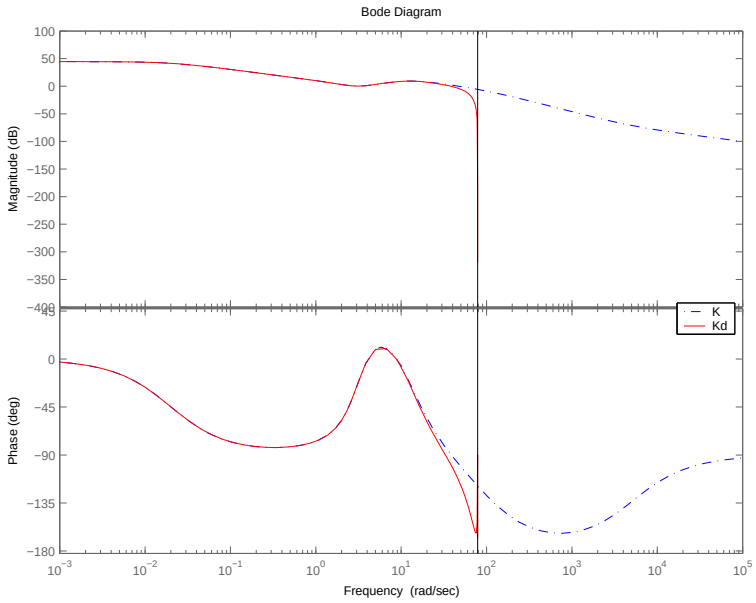


FIG. 4.10 – Réponse fréquentielle du correcteur K et de sa discrétisation K_d

D'après la figure 4.10 la bande passante du correcteur est de 63 rad/s soit environ 10 Hz. En appliquant le théorème de Shannon, la période d'échantillonnage maximale est 50 ms. Le temps de montée de la boucle fermée est de 0,34 s. Pour avoir entre 4 à 10 échantillons pendant ce temps de montée la période d'échantillonnage doit être comprise entre 34 et 85 ms. Pour ne pas réduire la bande passante du correcteur nous prenons une période d'échantillonnage inférieure à 50 ms. Nous choisissons arbitrairement 40 ms. Le correcteur à temps discret est obtenu par discrétisation approchée via la méthode de Tustin. Sa réponse fréquentielle, en pointillés sur la figure 4.10, est très proche de celle du correcteur à temps continu.

Il est certainement possible d'améliorer cette synthèse ou de choisir différemment la période d'échantillonnage, mais néanmoins cela permet d'avoir une base de comparaison. On retiendra de cette synthèse qu'il est possible d'obtenir la performance requise avec une période d'échantillonnage de 40 ms et les gabarits décrits précédemment.

4.4.3 Formulation du correcteur à période variable

Nous illustrons maintenant la conception d'un correcteur à période variable pour le système (4.38). La période h est comprise entre 40 et 120 ms.

Discrétisation du modèle du procédé

Dans le contexte de notre exemple, le fonctionnement nominal est défini pour la plus petite période. Il est alors judicieux de choisir $h_0 = h_{min} = 40$ ms comme localisation de l'approximation de Taylor.

Les expressions (4.15) et (4.16) sont utilisées pour calculer le modèle affine de la discrétisation paramétrée du système (4.38). L'erreur de discrétisation, selon le critère (4.17), est présentée sur la figure 4.11. Comme on peut s'y attendre, l'erreur diminue quand l'ordre N de la série augmente et quand la période h tend vers h_0.

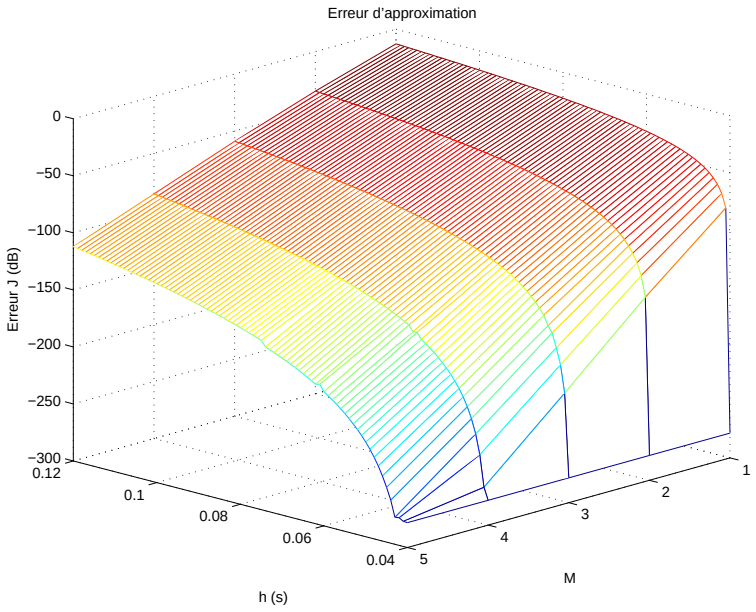


FIG. 4.11 – Erreur d'approximation

Ce tracé empirique permet de choisir l'ordre de la série pour garantir une borne supérieure de l'erreur d'approximation. Dans le cas de cet exemple l'ordre sera choisi plus tard car son

effet sera étudié lors de la synthèse du correcteur.

Caractéristique fréquentielle de l'erreur La figure 4.11 présente l'erreur au sens de la norme H_∞, c'est à dire dans le pire cas. Notre exemple étant mono-entrée/mono-sortie le tracé de la réponse fréquentielle de $G_d(z) - \tilde{G}_{d_M}(z)$ précise la répartition fréquentielle de l'erreur. Ce tracé est présenté sur la figure 4.12 pour 8 périodes comprises entre 40 et 120 ms et 4 valeurs de l'ordre M de la série.

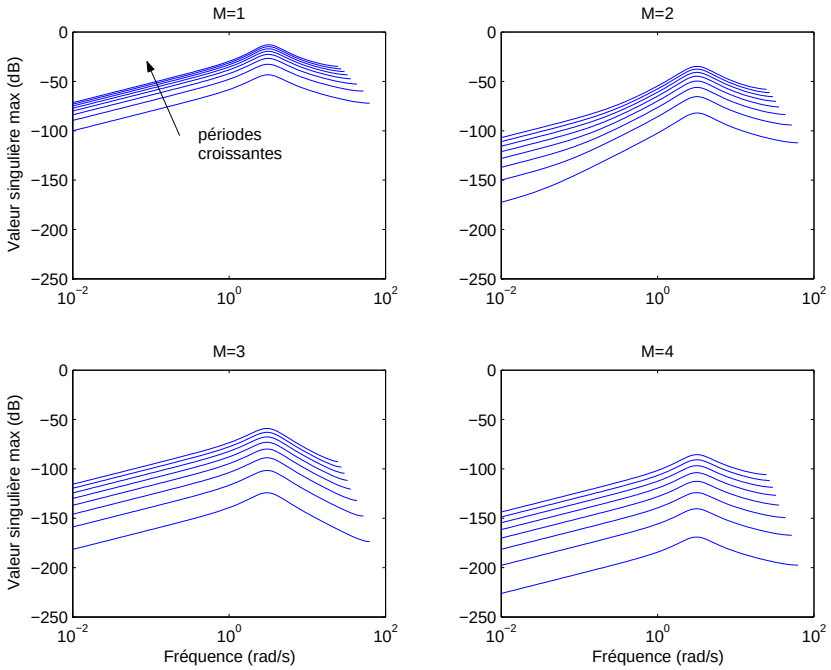


FIG. 4.12 – Répartition fréquentielle de l'erreur d'approximation

On observe une nouvelle fois une baisse de l'erreur quand M augmente. L'erreur est la plus grande pour les pulsations proches de la pulsation de résonance du système. Cela peut être problématique car un bon modèle aux fréquences caractéristiques du procédé est toujours préférable. Il apparaît aussi que la forme de la réponse fréquentielle n'est pas modifiée par la variation de l'ordre et de la période, elle subit simplement un décalage vertical. Il semble donc que les pôles et les zéros de cette réponse fréquentielle soient indépendants de la période et de l'ordre de la série. Seul le gain statique est modifié par leur variation. Si $h_0 = h_{min}$, le tracé de la réponse en fréquences pour h_{max} suffit à déterminer l'enveloppe fréquentielle de l'erreur.

Notons que cette enveloppe permet de modéliser, sous la forme d'une incertitude non paramétrique, l'erreur de discrétisation.

Discrétisation des gabarits fréquentiels

Après la discrétisation du modèle du procédé, il faut maintenant discrétiser les gabarits fréquentiels qui définissent les objectifs en boucle fermée.

Nous reprenons les gabarits de la synthèse de référence. En suivant la démarche du paragraphe 4.3, la partie constante est un gain statique unitaire et la partie variable est définie par :

$$W_S(p,f) = \frac{p\,M_S + \omega_S(f)}{p + \omega_S(f)\,\epsilon_S} \qquad W_{KS}(p,f) = \frac{p + \omega_{KS}(f)/M_{KS}}{\epsilon_{KS}\,p + \omega_{KS}(f)} \tag{4.43}$$

avec $M_S = 2$, $\epsilon_S = 0,005$, $M_{KS} = 20$ et $\epsilon_{KS} = 0,01$ et

$$\omega_S(f) = h_{min}\,\omega_{S_{max}}\,f \qquad \omega_{KS}(f) = h_{min}\,\omega_{KS_{max}}\,f \tag{4.44}$$

où $\omega_{S_{max}} = 4\,\text{rad/s}$, $\omega_{KS_{max}} = 50\,\text{rad/s}$ et $f \in [8,3;25]\text{Hz}$. Ainsi ces deux pulsations sont linéairement dépendantes de la fréquence d'échantillonnage et correspondent aux spécifications de la synthèse de référence quand $h = h_{min}$.

Gabarits en hautes fréquences et échantillonnage Les figures 4.13 et 4.14 présentent les gabarits fréquentiels $1/W_S$ et $1/W_{KS}$ évalués pour 5 périodes d'échantillonnage entre 40 et 120 ms. Les gabarits à temps continu, dont la pulsation est une fonction du paramètre f, sont en bleu (pointillés) et leur discrétsation en rouge (trait plein).

Le gabarit $1/W_S(z)$ à temps discret représente fidèlement la spécification à temps continu à une petite variation près du maximum M_S.

Le gabarit $1/W_{KS}(z)$ ne reproduit pas la spécification à temps continu. Ceci est dû à la présence d'un pôle dans $W_{KS}(p)$ dont la pulsation est supérieure à la pulsation d'échantillonnage. Cela conduit à une translation du gabarit vers les basses fréquences.

Une première solution à ce problème est de simplifier le gabarit $1/W_{KS}$ en supprimant le zéro en hautes fréquences qui ne paraît pas utile à première vue. Malheureusement, dans ce cas le transfert W_{KS} ne serait plus causal car il aurait un zéro et aucun pôle. Cette solution est donc inapplicable.

Une deuxième solution est d'augmenter la pulsation d'échantillonnage pour qu'elle soit supérieure à celle du pôle. Cela conduit à sur-échantillonner le système et à ne plus respecter le cahier des charges.

Une dernière solution est d'augmenter l'ordre du gabarit pour que la pente soit plus forte et que le pôle de $W_{KS}(p)$ ait une pulsation inférieure à la pulsation d'échantillonnage. Le risque dans ce cas est d'augmenter inutilement l'ordre de ce gabarit, ce qui accroît celui du système augmenté et donc celui du correcteur synthétisé.

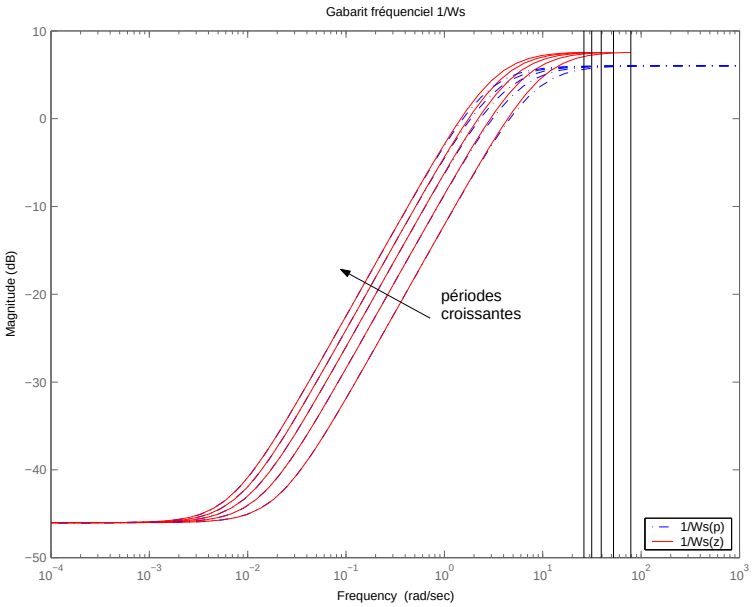


FIG. 4.13 – Réponse fréquentielle de la discrétisation du gabarit sur S

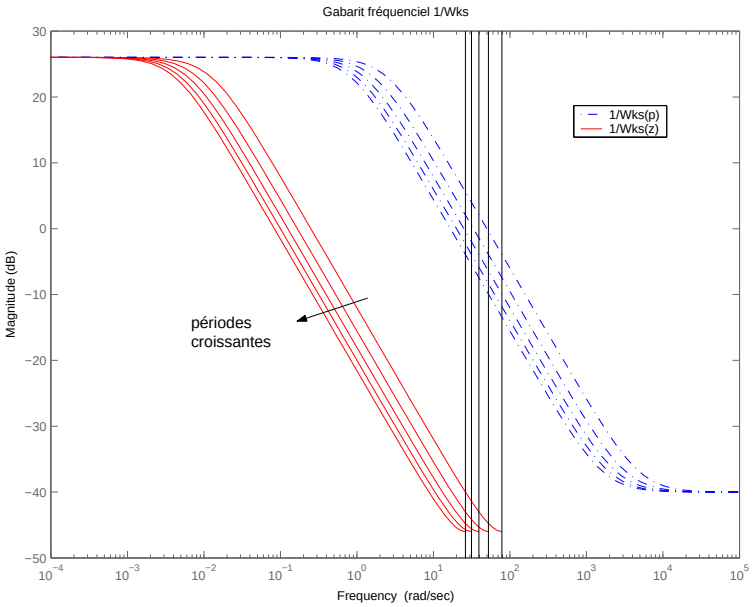


FIG. 4.14 – Réponse fréquentielle de la discrétisation du gabarit sur KS

Cet exemple montre qu'il est difficile d'ajuster la sensibilité KS en hautes fréquences tout en limitant la période d'échantillonnage et l'ordre du correcteur. Sachant que contraindre le gain de KS en hautes fréquences limite la bande passante du correcteur, il est donc difficile de réduire sa sensibilité aux bruits.

Dans le cas présent, le gabarit $1/W_{KS}$ obtenu après discrétisation a une bande passante comprise entre 0,11 et 0,35 rad/s, qui est beaucoup trop faible. La contrainte en hautes fréquences est trop forte sur la commande et la synthèse produit un correcteur très loin des objectifs demandés. Nous proposons alors de simplifier le gabarit de KS par un gain statique, tout en sachant que le correcteur devient sensible aux bruits en hautes fréquences. On définit alors :

$$W_{KS} = \frac{1}{M_{KS}} \tag{4.45}$$

Construction du système augmenté

Avant d'appliquer les théorèmes de synthèse, il faut construire le système augmenté composé des modèles à temps discret du procédé et des gabarits.

Exemple Nous illustrons ici cette construction pour une série de Taylor d'ordre $M = 2$ et un polytope dont le nombre de sommets est réduit. Le système augmenté est alors défini comme la combinaison convexe de $M + 1 = 3$ systèmes sommets :

$$\mathcal{P}'(H) = \begin{pmatrix} \mathcal{A}'(H) & \mathcal{B}'(H) \\ \mathcal{C}' & \mathcal{D}' \end{pmatrix} = \sum_{i=1}^{3} \alpha_i P_i = \begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix} \tag{4.46}$$

avec, conformément à l'expression (4.36) :

$$\mathcal{B}'(H) = \begin{pmatrix} \mathcal{B}'_w(H) & \mathcal{B}'_u(H) \end{pmatrix} \qquad \mathcal{C}' = \begin{pmatrix} C'_z \\ C'_y \end{pmatrix} \qquad \mathcal{D}' = \begin{pmatrix} D'_{zw} & D'_{zu} \\ D'_{yw} & D'_{yu} \end{pmatrix} \tag{4.47}$$

Les trois systèmes sont définis par :

$$A_1 = \begin{pmatrix} 0.9925 & 0.0376 & 0 \\ -0.3688 & 0.8797 & 0 \\ -0.1595 & 0 & 0.9992 \end{pmatrix} \quad A_2 = \begin{pmatrix} 0.9630 & 0.1080 & 0 \\ -1.0592 & 0.6391 & 0 \\ -0.1595 & 0 & 0.9992 \end{pmatrix} \quad A_3 = \begin{pmatrix} 0.9353 & 0.0983 & 0 \\ -0.9648 & 0.6403 & 0 \\ -0.1595 & 0 & 0.9992 \end{pmatrix} \tag{4.48}$$

$$B_1 = \begin{pmatrix} 0 & 0.0075 \\ 0 & 0.3688 \\ 0.1595 & 0 \end{pmatrix} \quad B_2 = \begin{pmatrix} 0 & 0.0370 \\ 0 & 1.0592 \\ 0.1595 & 0 \end{pmatrix} \quad B_3 = \begin{pmatrix} 0 & 0.0647 \\ 0 & 0.9648 \\ 0.1595 & 0 \end{pmatrix} \tag{4.49}$$

$$C_1 = \begin{pmatrix} -0.5000 & 0 & 1.0000 \\ 0 & 0 & 0 \\ -1.0000 & 0 & 0 \end{pmatrix} \quad C_2 = \begin{pmatrix} -0.5000 & 0 & 1.0000 \\ 0 & 0 & 0 \\ -1.0000 & 0 & 0 \end{pmatrix} \quad C_3 = \begin{pmatrix} -0.5000 & 0 & 1.0000 \\ 0 & 0 & 0 \\ -1.0000 & 0 & 0 \end{pmatrix} \tag{4.50}$$

$$D_1 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0.0500 \\ 1.0000 & 0 \end{pmatrix} \quad D_2 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0.0500 \\ 1.0000 & 0 \end{pmatrix} \quad D_3 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0.0500 \\ 1.0000 & 0 \end{pmatrix} \tag{4.51}$$

L'ajout d'un filtre passe-bas sur la commande, pour respecter l'hypothèse H_2 nécessaire à la synthèse, modifie le système augmenté qui devient d'ordre 4. Les 3 systèmes sont alors définis par les matrices suivantes où la matrice $\mathcal{B}'_u$, définie par la combinaison des blocs en gris, devient indépendante de H.

$$A_1 = \begin{pmatrix} 0.9925 & 0.0376 & 0 & 0.0075 \\ -0.3688 & 0.8797 & 0 & 0.3688 \\ -0.1595 & 0 & 0.9992 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A_2 = \begin{pmatrix} 0.9630 & 0.1080 & 0 & 0.0370 \\ -1.0592 & 0.6391 & 0 & 1.0592 \\ -0.1595 & 0 & 0.9992 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A_3 = \begin{pmatrix} 0.9353 & 0.0983 & 0 & 0.0647 \\ -0.9648 & 0.6403 & 0 & 0.9648 \\ -0.1595 & 0 & 0.9992 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.52}$$

$$B_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0.1595 & 0 \\ 0 & 1.0000 \end{pmatrix} \quad B_2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0.1595 & 0 \\ 0 & 1.0000 \end{pmatrix} \quad B_3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0.1595 & 0 \\ 0 & 1.0000 \end{pmatrix} \tag{4.53}$$

$$C_1 = \begin{pmatrix} -0.5000 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0.0500 \\ -1.0000 & 0 & 0 & 0 \end{pmatrix} \quad C_2 = \begin{pmatrix} -0.5000 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0.0500 \\ -1.0000 & 0 & 0 & 0 \end{pmatrix} \quad C_3 = \begin{pmatrix} -0.5000 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0.0500 \\ -1.0000 & 0 & 0 & 0 \end{pmatrix} \tag{4.54}$$

$$D_1 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0 \\ 1.0000 & 0 \end{pmatrix} \quad D_2 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0 \\ 1.0000 & 0 \end{pmatrix} \quad D_3 = \begin{pmatrix} 0.5000 & 0 \\ 0 & 0 \\ 1.0000 & 0 \end{pmatrix} \tag{4.55}$$

4.4.4 Synthèse du correcteur à période variable

Comparaison des différentes synthèses

Nous profitons de cet exemple pour comparer l'influence sur la synthèse des trois paramètres suivants :

- L'ordre M de la série de Taylor
- L'usage d'un polytope complet (2^M sommets) ou réduit ($M+1$ sommets)
- Le méthodologie de synthèse avec une fonction de Lyapunov constante (théorème 1) ou dépendante de la période (théorème 2).

Notons qu'avant d'appliquer une synthèse nous vérifions le respect des hypothèses H_1, H_2 et H_3. Les 4 synthèses possibles sont définies selon le tableau suivant :

Méthode	Matrice de Lyapunov	Polytope
A	constante	complet
B	constante	réduit
C	polytopique	complet
D	polytopique	réduit

TAB. 4.1 – Combinaison des synthèses possibles

Les résultats des différentes synthèses sont regroupés dans les tableaux 4.2 où γ est la borne obtenue pour le problème de sensibilité mixte de la figure 4.8 dans sa version à temps discret.

(a) M=1

Méthode	Sommets	γ
A	2	1.09
B	2	1.09
C	2	1.53
D	2	1.54

(b) M=2

Méthode	Sommets	γ
A	4	1.27
B	3	1.15
C	4	1.58
D	3	1.55

(c) M=3

Méthode	Sommets	γ
A	8	1.33
B	4	1.15
C	8	1.64
D	4	1.55

(d) M=4

Méthode	Sommets	γ
A	16	1.33
B	5	1.15
C	16	1.64
D	5	1.55

TAB. 4.2 – Résultats des différentes synthèses

On constate premièrement qu'une augmentation de l'ordre de la série de Taylor induit une augmentation de γ. Néanmoins la très faible variation de γ entre $M = 3$ et $M = 4$ laisse penser que γ tend vers une valeur limite. Cela peut s'expliquer par un apport négligeable d'information au delà d'un certain ordre.

Deuxièmement la réduction du nombre de sommets du polytope diminue la valeur de γ. Ceci s'illustre entre les méthodes A et B de même que C et D. Ce résultat peut s'expliquer par la réduction du conservatisme dû à l'approximation grossière d'une courbe par un ensemble convexe.

Troisièmement l'usage d'une matrice de Lyapunov dépendant du paramètre n'améliore pas la synthèse. Plusieurs hypothèses peuvent expliquer cette observation inattendue. D'une part l'exemple est peut-être trop simple pour qu'une différence soit perceptible. D'autre part l'implémentation numérique des méthodes C et D n'est pas aussi optimisée que celle des méthodes A et B. En effet la méthode du théorème 1 est issue d'une version modifiée de la boîte à outils LMI de matlab, optimisée par ces auteurs, alors que nous avons programmé les méthodes C et D sans optimisation numérique.

A la lecture de ces tableaux, l'ordre $M = 1$ est le meilleur choix vis à vis du respect des objectifs de la synthèse. Néanmoins, sachant que la synthèse est basée sur le modèle LPV à temps discret, la valeur de γ ne prend pas en compte l'erreur d'approximation lors de la construction de ce modèle. Ainsi pour $M = 1$ la synthèse satisfait très bien les objectifs, mais sur un modèle très approximatif. Il y a donc un compromis à trouver pour avoir un modèle suffisamment représentatif sans trop pénaliser la synthèse.

Notons que la reconstruction en ligne du correcteur nécessite la combinaison convexe d'un nombre de matrices égal au nombre de sommets du polytope. Le choix de M influence alors

la complexité de l'implémentation du correcteur. Il est donc judicieux d'intégrer ce point dans le compromis entre la précision du modèle et la complexité de la synthèse.

Remarque 15 *Pour éviter l'insensibilité de γ vis à vis de l'erreur d'approximation, il faudrait intégrer cette dernière dans la synthèse. Une solution est de modéliser cette erreur par une incertitude non paramétrique et de réaliser une synthèse LPV robuste. Malheureusement, l'approche polytopique utilisée ici ne permet pas d'intégrer des incertitudes dans la synthèse. Cette limite de l'approche polytopique a motivé la recherche d'une autre formulation et notamment celle de la fin de ce chapitre qui repose sur l'utilisation de la représentation linéaire fractionnelle.*

Conclusion de la comparaison On constate premièrement qu'un ordre réduit de la série de Taylor est préférable pour le respect des objectifs de la synthèse. Néanmoins le choix de M est un compromis entre la précision du modèle, la complexité de la synthèse et la complexité de la reconstruction du correcteur. Deuxièmement la réduction du nombre de sommets du polytope est bénéfique pour la synthèse et pour la reconstruction du correcteur. Enfin il est difficile de conclure sur l'intérêt d'utiliser une fonction de Lyapunov dépendant de la période.

Détail d'une synthèse

Pour cette étude détaillée, l'ordre de la série vaut $M = 2$, le polytope est réduit et la matrice de Lyapunov est constante. Dans ce cas, d'après la figure 4.12, l'erreur d'approximation du modèle est inférieure à -35 dB.

Les figure 4.15 à 4.16 présentent les fonctions de sensibilité obtenues à l'issue de la synthèse. Les courbes rouges (trait plein) représentent les gabarits définis par les équations (4.43) et (4.45). Les courbes bleues (pointillés) représentent les fonctions de sensibilité évaluées, pour plusieurs valeurs de la période, sur l'interconnexion entre le correcteur discret et le procédé discrétisé à la valeur de la période. Enfin la courbe verte représente la fonction de sensibilité de la synthèse de référence.

Sur la figure 4.15, on constate une erreur statique inférieure à -45 dB, un pic de sensibilité de S variant de 0,8 à 3 dB et une pulsation ω_S comprise entre 1,2 et 3 rad/s quand h varie de 120 à 40 ms. En comparaison avec la synthèse de référence, l'erreur statique est similaire, la pulsation de S varie dans un rapport de 2,5 et le pic de S est parfois supérieur.

Sur la figure 4.16, le maximum de KS est compris entre 12 et 25 dB. Comme énoncé lors de la discrétisation des gabarits, l'absence d'une contrainte en hautes fréquences sur W_{KS} conduit à une forte sensibilité de la commande vis à vis des bruits en hautes fréquences.

La figure 4.17 montre la boucle fermée dont la bande passante varie de 1,4 à 7 rad/s. Elle est donc légèrement plus rapide que celle de la synthèse de référence pour les périodes les plus petites.

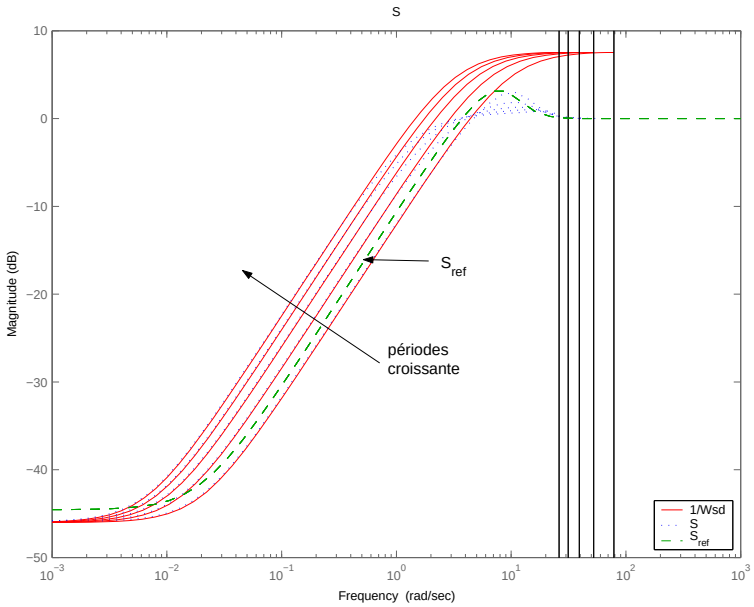


FIG. 4.15 – Réponse fréquentielle de la sensibilité S

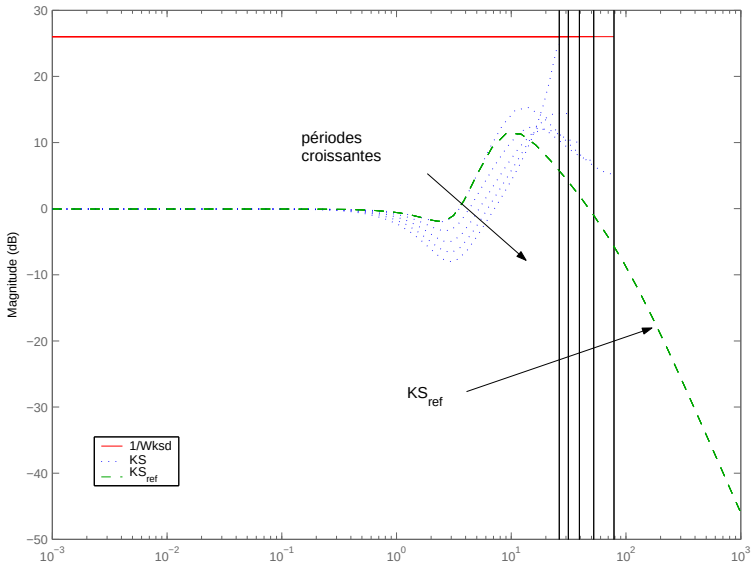


FIG. 4.16 – Réponse fréquentielle de la sensibilité KS

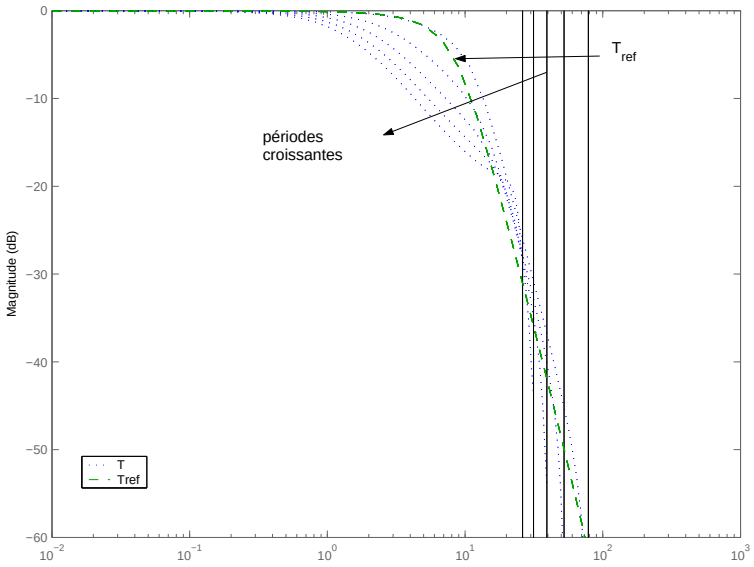


FIG. 4.17 – Réponse fréquentielle de la sensibilité T

Au vu de ces courbes, les sensibilités de la boucle fermée évoluent dans des proportions similaires à la variation de la période. Nous étudions maintenant, en simulation, la réponse temporelle de la boucle fermée.

4.4.5 Simulations de la boucle fermée

Les simulations de la boucle fermée sont réalisées avec le modèle linéaire de l'équation (4.38).

Réponse indicielle

Dans un premier temps nous étudions la réponse temporelle de la boucle fermée pour 5 valeurs de la période comprises entre 40 et 120 ms. La figure 4.18 présente la réponse indicielle de l'interconnexion entre le correcteur à temps discret et le procédé à temps continu.

Le temps de réponse à 5% varie entre 0,46 et 2,22 s, soit dans un rapport de 4,8. Pour les plus petites valeurs de la période, il est même plus faible que celui de la synthèse de référence, ce qui est cohérent avec la remarque sur la sensibilité T. Le dépassement est nul et l'erreur statique est inférieure à 0,01, ce qui correspond aux objectifs recherchés.

Le comportement que nous observons correspond parfaitement à nos attentes et bien que le temps de réponse varie dans un rapport de 4,8 au lieu de 3 le comportement face à l'augmentation de la période est progressif et prévisible.

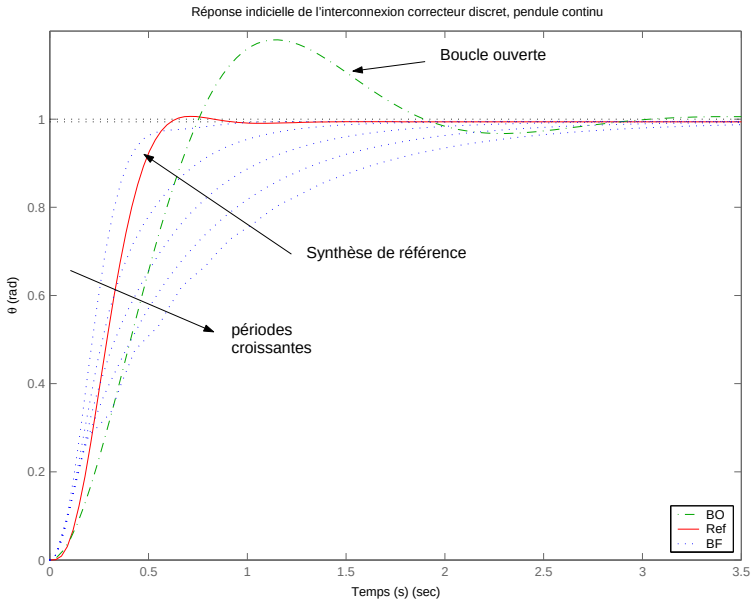


FIG. 4.18 – Réponse indicielle du modèle à temps continu

Variation en ligne de la période

Les réponses temporelles de la figure 4.18 sont obtenues pour différentes valeurs de la période, mais maintenues constantes pendant la durée de l'échelon. Nous étudions maintenant le comportement du système en boucle fermée face à des variations de la période d'échantillonnage qui interviennent à tout moment.

Dans le scénario de la figure 4.19, la période d'échantillonnage évolue selon un signal sinusoïdal de pulsation 0,5 rad/s et d'amplitude comprise entre 40 et 120 ms. La référence angulaire du pendule est excitée par un signal carré de période 6 s et de rapport cyclique 50 %.

Conformément aux résultats de la synthèse, plus la période est petite et plus le temps de réponse est court. Ainsi lors du premier créneau descendant, entre 3 et 6 seconde, la boucle fermée est lente alors que la période est presque à son maximum. A l'inverse lors du deuxième créneau descendant entre 9 et 12 seconde, le système en boucle fermée est beaucoup plus rapide car la période est à son minimum. Le comportement de la commande est aussi affecté par la valeur de la période. Ainsi elle plus oscillante quand la période est grande. Ceci peut s'expliquer par l'évolution plus importante du système entre deux échantillonnages quand la période est élevée, nécessitant une commande plus forte pour le corriger.

Sur la figure 4.20 la référence angulaire du pendule reste identique mais la variation de la période est beaucoup plus brutale puisqu'elle passe toutes les 7 secondes de 40 à 120 ms et inversement.

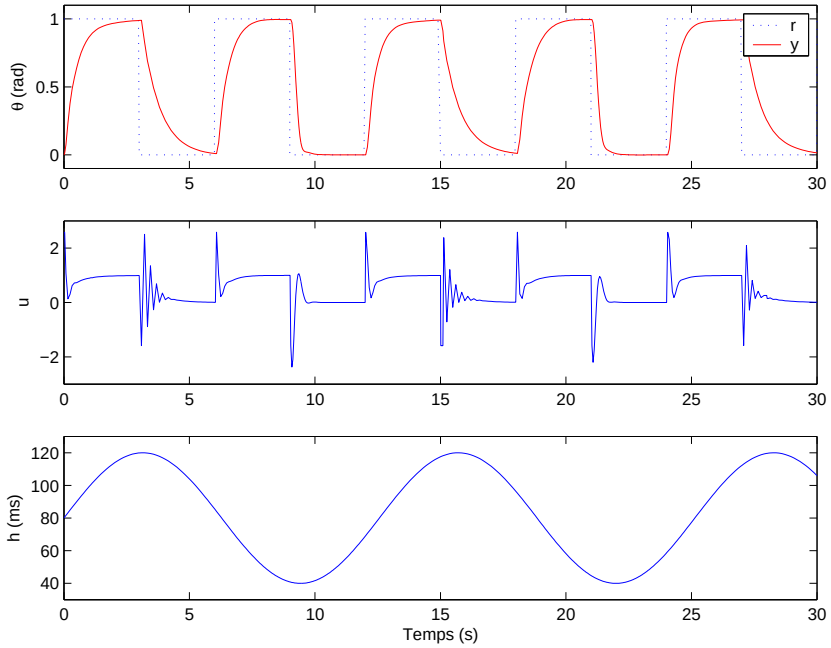


FIG. 4.19 – Evolution du pendule face à une période sinusoïdale

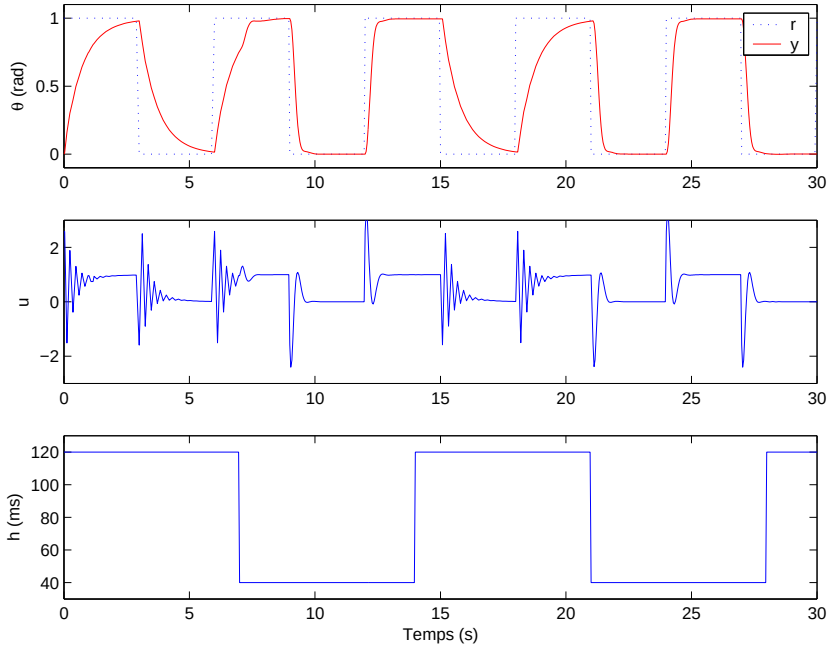


FIG. 4.20 – Evolution du pendule face à une période en créneaux

Le temps de réponse du pendule évolue une nouvelle fois de façon cohérente avec la valeur de la période. Au cours du deuxième créneau montant de la référence r, la réponse de la boucle fermée s'accélère brusquement quand à 7 secondes, la période passe du maximum au minimum. Ce changement brutal de période ne provoque de perturbations ni sur la sortie y ni sur la commande u. Cela illustre le grand intérêt d'utiliser une synthèse LPV. En effet la stabilité est garantie par la synthèse, quelle que soit la trajectoire d'évolution du paramètre variable, c'est à dire dans notre cas, quelle que soit l'évolution de la période dans l'intervalle 40 à 120 ms. Notons qu'aux différents changements de la période, à 14, 21 et 28 seconde, aucune variation, aucun pic ne sont visibles sur la commande, l'actionneur n'est donc pas brutalisé. Comme dans le scénario précédent, la commande est plus oscillante quand la période est grande.

Conclusion des simulations Ces différentes simulations montrent une boucle fermée conforme à nos attentes et dont le comportement est prévisible. D'une part les oscillations du procédé sont complètement atténuées et le dépassement est maintenant nul. D'autre part le temps de réponse évolue linéairement avec la période et le procédé est même légèrement plus rapide que la synthèse de référence pour les plus petites périodes. Enfin la commande reste douce quelle que soit la vitesse de variation de la période ; l'actionneur n'est donc pas brutalisé.

4.4.6 Comparaison avec une synthèse à gabarits constants

Nous avons proposé, au paragraphe 4.3.1, d'ajuster les objectifs de la boucle fermée à la variation de la période d'échantillonnage. Sachant qu'il y a une tolérance dans le choix de cette période, on peut s'interroger sur la nécessité d'adapter les objectifs. Par exemple, le critère de l'expression (4.9) préconise une période qui peut varier d'un facteur 3. Pour tenter de répondre à cette question, on propose de comparer deux correcteurs, l'un avec des objectifs adaptés à la période, l'autre avec des objectifs constants.

Pour construire le correcteur à objectifs constants, on reprend le système augmenté de la synthèse de référence. Il est composé des modèles à temps continu du procédé et des gabarits fréquentiels. Sachant que les gabarits sont constants, il n'est pas nécessaire de les discrétiser à part, c'est pourquoi le système augmenté est construit avec les modèles à temps continu puis discrétisé selon la méthode du paragraphe 4.2.

Nous utilisons une série de Taylor d'ordre 2 et le polytope est réduit pour avoir 3 sommets. A l'issue de la synthèse, avec une fonction de Lyapunov constante, γ vaut 1,44. Les fonctions de sensibilité sont présentées sur les figures 4.21 à 4.23

A la lecture de ces 3 figures, on observe sur la fonction de sensibilité S un pic compris entre 0,5 et 1,8 dB, une pulsation comprise entre 2,6 et 3,1 rad/s et une erreur statique inférieure à -44,7 dB. La bande passante de la boucle fermée T varie de 2,7 à 4,3 rad/s. Enfin

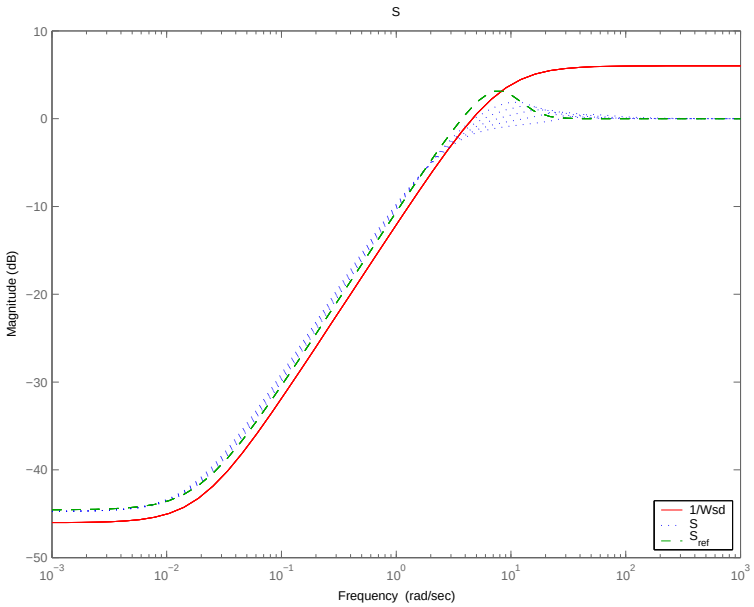


FIG. 4.21 – Réponse fréquentielle de la fonction de sensibilité S

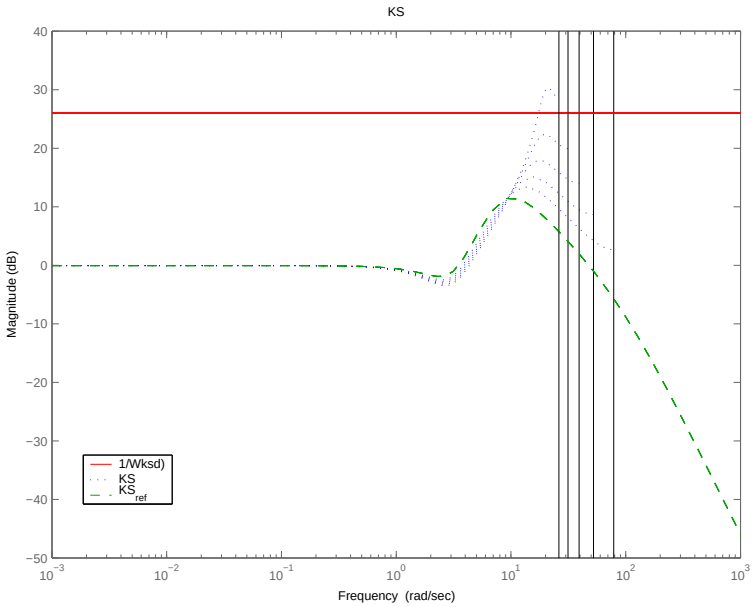


FIG. 4.22 – Réponse fréquentielle de la fonction de sensibilité KS

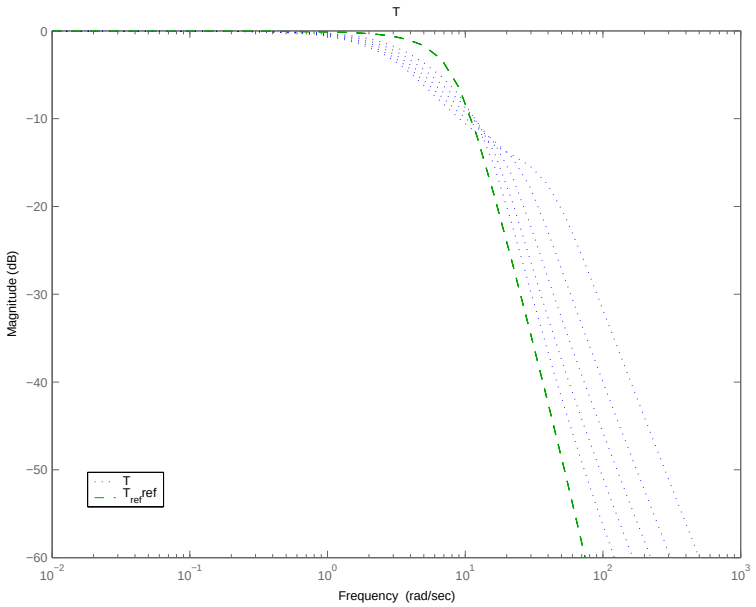


FIG. 4.23 – Réponse fréquentielle de la fonction de sensibilité T

le maximum de la sensibilité KS évolue entre 13 et 30 dB.

De ces observations, on retient que les fonctions de sensibilité varient légèrement en fonction de h alors que les objectifs sont constants. Vis à vis de la synthèse de référence, la sensibilité S a un maximum plus faible, les pulsations de S et T varient légèrement et le pic de KS est un peu plus élevé. On peut alors s'attendre à une boucle fermée moins rapide et une commande plus sensible aux bruits en hautes fréquences.

Sur la figure 4.24, la réponse indicielle de la boucle fermée est évaluée pour 5 périodes d'échantillonnage comprises entre 40 et 120 ms. Le temps de réponse varie entre 0,7 et 1,1 s. La boucle fermée est donc plus rapide que la boucle ouverte mais plus lente que la synthèse de référence. De plus son temps de réponse évolue très peu en fonction de la période. Ce comportement temporel est finalement cohérent avec les fonctions de sensibilité.

Pour terminer cette comparaison nous simulons les deux scénarios où la période évolue au cours du temps. Les résultats sont illustrés sur les figures 4.25 et 4.26.

Comme on pouvait s'y attendre, le temps de réponse varie très faiblement en fonction de la période. Les changements rapides de la période ne provoquent pas de "secousses" sur la commande. Par contre l'amplitude de cette dernière est plus forte avec le correcteur à objectifs constants qu'avec celui à objectifs variables, ce qui est cohérent avec la différence entre leur sensibilité KS respective. Enfin la commande est plus oscillante quand la période est grande.

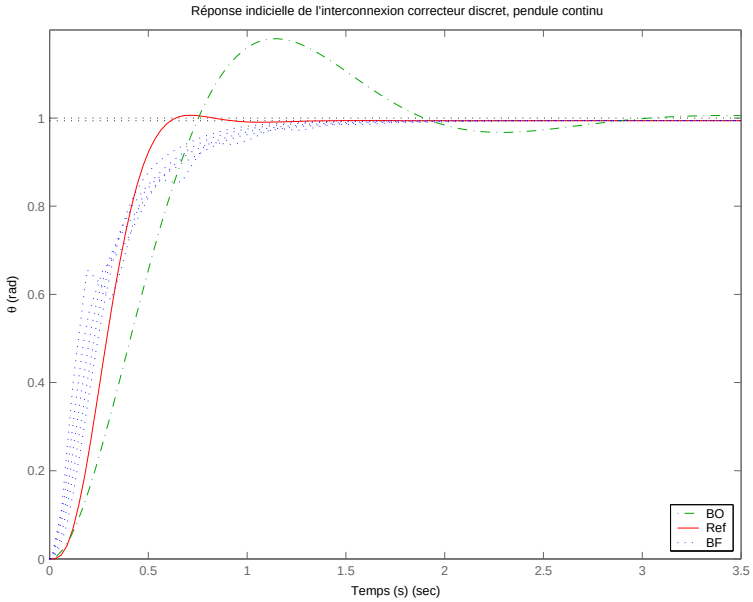


FIG. 4.24 – Réponse indicielle du modèle à temps continu

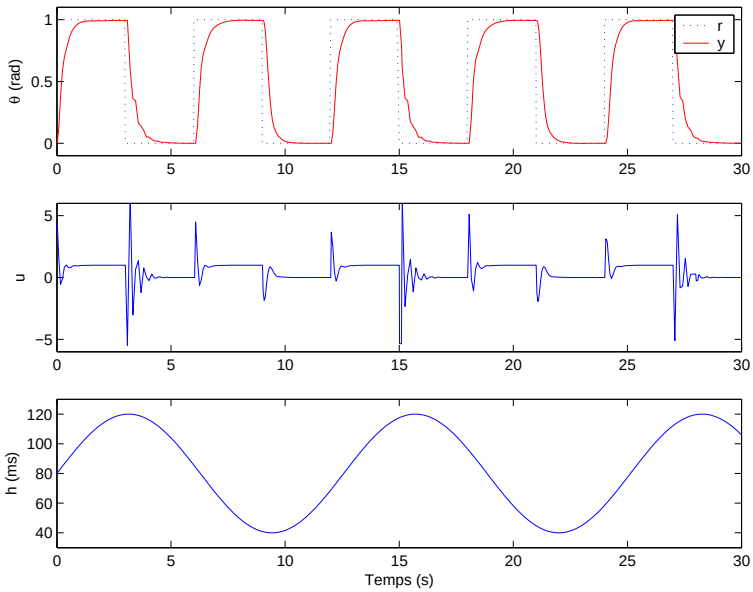


FIG. 4.25 – Evolution du pendule face à une période sinusoïdale

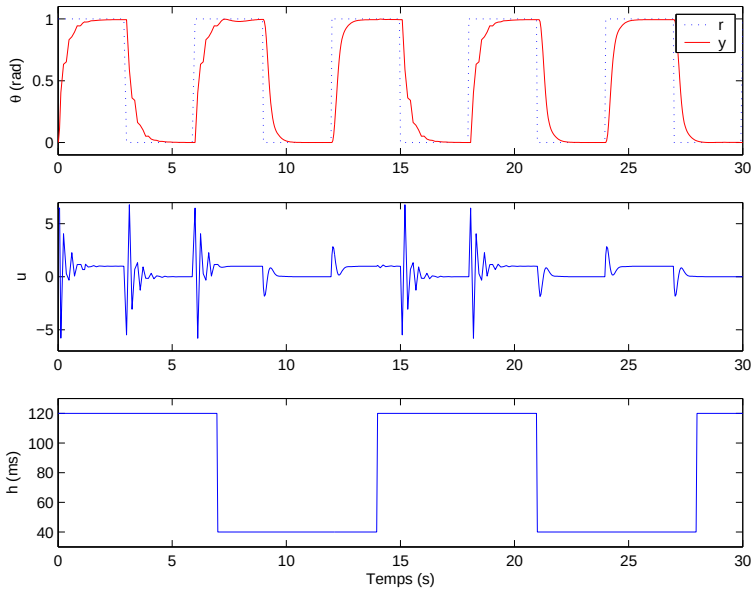


FIG. 4.26 – Evolution du pendule face à une période en créneaux

Conclusion de cette comparaison L'objectif de cette comparaison est d'évaluer la nécessité d'adapter la performance du correcteur à la variation de la période. Dans le cas de cet exemple, les objectifs constants semblent envisageables mais au détriment d'une commande dont l'amplitude est plus forte.

On constate, avec les objectifs constants, une boucle fermée plus lente que celle de la synthèse de référence. A l'inverse, avec des objectifs variables, la boucle fermée peut être plus rapide que celle de la synthèse de référence. Il semble qu'il y ait un compromis entre adapter les objectifs, tout en pouvant atteindre leur maximum quand la période est la plus favorable, et les maintenir constants mais à un niveau inférieur à leur maximum.

La robustesse des deux correcteurs n'a pas été étudiée. Néanmoins il est probable que le maintient de performances élevées quand la période est grande soit au détriment de la robustesse. C'est une piste qu'il faudra étudier dans le futur.

Cette comparaison a été réalisée sur le modèle, très simple, de l'expression (4.38) dans la continuité de ce chapitre. Malheureusement les résultats obtenus sont insuffisants pour dresser des conclusions objectives. Il semble nécessaire, dans le futur, de refaire la démarche avec un système plus complexe, un intervalle de périodes plus grand ou des objectifs plus difficiles.

4.5 Autre formulation : modélisation LFR

Nous présentons maintenant une seconde formulation pour concevoir un correcteur à période variable. Dans l'introduction des systèmes LPV du chapitre 3, deux familles d'approches pour le contrôle des systèmes LPV ont été mentionnées. La méthodologie présentée jusqu'à présent utilise un modèle polytopique et une fonction de Lyapunov quadratique en l'état. Nous formulons maintenant notre problème pour la seconde famille qui repose sur la transformée linéaire fractionnelle (LFT) et le théorème du petit gain.

A l'écriture des ces lignes, cette approche ne donne pas de résultats numériques probants malgré une formulation qui nous semble intéressante.

4.5.1 Rappels et notations

La transformation linéaire fractionnelle (LFT) joue un rôle essentiel dans la représentation des incertitudes. Elle étend les notions de fonction de transfert et de représentation d'état en y ajoutant des incertitudes de nature fréquentielle ou paramétrique. Nous rappelons ici les bases de la transformation LFT, extraites du livre de Skogestad and Postlethwaite (1996).

Soit une matrice P de dimension $(n_1 + n_2) \times (m_1 + m_2)$ et partitionnée comme suit :

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \tag{4.56}$$

Soit deux matrices Δ et K de dimension $m_1 \times n_1$ et $m_2 \times n_2$, les transformations linéaires fractionnelles supérieure et inférieure sont respectivement définies par :

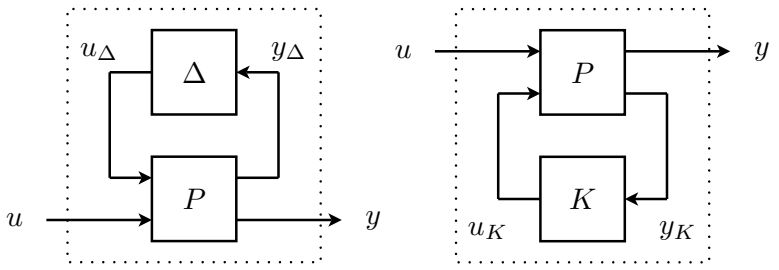
$$F_u(P, \Delta) := P_{22} + P_{21}\Delta(I - P_{11}\Delta)^{-1}P_{12} \tag{4.57}$$

$$F_l(P, K) := P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} \tag{4.58}$$

Ces transformations sont équivalentes aux interconnexions de la figure 4.27. L'interconnexion (a) est traditionnellement utilisée pour représenter un système incertain où Δ modélise les incertitudes. La structure de Δ représente la structure des incertitudes. L'interconnexion (b) illustre l'interconnexion entre un procédé P et un correcteur K. Elle est classiquement utilisée dans la synthèse de correcteurs.

Les expressions (4.57) et (4.58) sont statiques. En remarquant qu'une fonction de transfert est une LFT :

$$\forall p \in C^*, F_u\left(\begin{pmatrix} A & B \\ C & D \end{pmatrix}, \frac{1}{p}I\right) = D + C(pI - A)^{-1}B := M(p) \tag{4.59}$$



(a) Supérieure (b) Inférieure

FIG. 4.27 – Transformation linéaire fractionnelle

on peut justifier des expressions de la forme :

$$F_u\left(M(p),\Delta\right) = F_u\left(F_u(M,\frac{1}{p}I),\Delta\right) \tag{4.60}$$

Il en est de même pour Δ et K qui peuvent être :

- Un opérateur linéaire stationaire (LTI) sans mémoire : Δ,
- Un opérateur linéaire à mémoire (LTI) : $\Delta(p)$,
- Un opérateur variant dans le temps (LTV) sans mémoire : $\Delta(t)$.

L'un des avantages de la LFT est la souplesse apportée dans la construction des interconnexions qui se résume par les propriétés essentielles suivantes :

- La somme de n LFT est une LFT.
- Le produit (mise en cascade) de n LFT est une LFT.
- L'inverse d'une LFT, quand elle existe, est une LFT.

Initialement utilisée pour étudier les systèmes incertains, la représentation linéaire fractionnelle (LFR) a été appliquée au contrôle des systèmes LPV par Packard (1994). Nous introduisons maintenant cette approche qui a motivé la formulation sous forme LFR de notre problème.

Approche LPV par modélisation LFR

Ce paragraphe est un bref résumé de l'article de Apkarian and Adams (1998) appliqué aux systèmes à temps discret.

L'objectif est illustré sur la figure 4.28. Le système à paramètres variants est décrit par la LFT $F_u(P(z),\Delta)$. La matrice Δ contient les paramètres dépendant du temps. Quand ces paramètres sont mesurables on peut chercher un correcteur sous la forme $F_l(K(z),\Delta)$, c'est à dire dépendant des mêmes paramètres que P.

La figure 4.29 présente une reformulation de la figure 4.28 avec les paramètres variants

regroupés dans le même bloc. Ainsi le problème original, l'interconnexion de deux systèmes LVP $P(z,\Delta)$ et $K(z,\Delta)$, est vue comme un problème de performance robuste, de l'interconnexion de deux systèmes LTI $P'(z)$ et $K(z)$, face à un bloc d'incertitudes défini par :

$$\Theta(t) = \begin{pmatrix} \Delta(t) & 0 \\ 0 & \Delta(t) \end{pmatrix} \in \mathbf{\Theta} \tag{4.61}$$

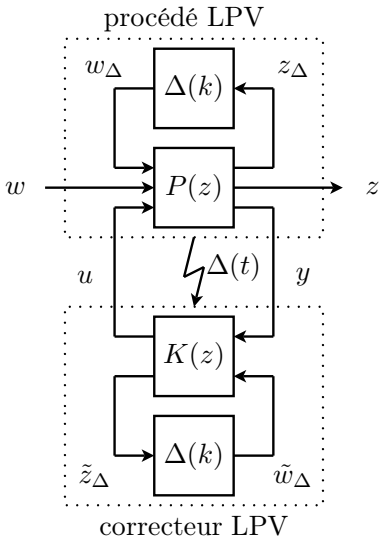


FIG. 4.28 – Structure de contrôle

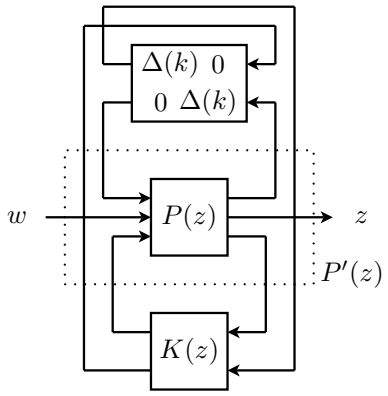


FIG. 4.29 – Structure de contrôle

Ainsi ce problème, similaire à la synthèse d'un correcteur LTI robuste, est résumé par le théorème suivant :

Théorème 3 (Condition de faisabilité)

Soit l'incertitude $\Theta(t) \in \mathbf{\Theta}$ normalisée telle que $\gamma^2\Delta^T\Delta \leq 1$. Soit une matrice de mise à l'échelle $L \in \{L > 0 : L\Theta = \Theta L, \forall\Theta \in \mathbf{\Theta}\}$.

S'il existe une matrice de mise à l'échelle L et un correcteur LTI $K(z)$ tels que l'interconnexion nominale $F_l(P'(z), K(z))$ est stable de manière interne et l'inégalité suivante est satisfaite :

$$\left\| \begin{pmatrix} L^{1/2} & 0 \\ 0 & I \end{pmatrix} F_l(P'(z), K(z)) \begin{pmatrix} L^{-1/2} & 0 \\ 0 & I \end{pmatrix} \right\|_\infty < \gamma \tag{4.62}$$

alors le correcteur à paramètres variants $F_l(K(z),\Delta)$ est le correcteur γ-optimal tel que, pour tout $\Delta(t)$, la boucle fermée $T_{zw} = F_l(F_u(P(z),\Delta), F_l(K(z),\Delta))$

- *est stable de manière interne*
- *a une norme L_2-induite inférieure à γ*

La recherche de la matrice L et du correcteur $K(z)$ est formalisée sous la forme d'inégalités matricielles linéaires à vérifier. Pour plus de détails, le lecteur peut se référer à l'article de Apkarian and Adams (1998).

Pour appliquer cette méthode à notre problème il faut modéliser sous la forme d'une LFT $F_l(P(z), \Delta)$ la discrétisation paramétrée d'un procédé continu. Nous allons maintenant décrire cette étape.

4.5.2 Construction du modèle LFT

L'objectif est de décrire sous la forme LFT de la figure 4.30, définie par (4.63), la discrétisation du modèle (4.1) pour une période h inconnue mais bornée par $h_{min} \leq h \leq h_{max}$. Nous présentons ici deux approches, l'une exacte, l'autre approchée.

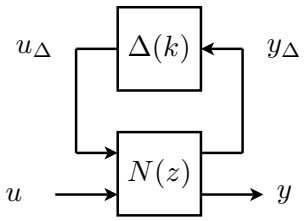


FIG. 4.30 – Forme LFT recherchée

$$N(z) := \begin{cases} x_{k+1} = \mathcal{A}x_k + \mathcal{B}\tilde{u}_k \\ \tilde{y}_k = \mathcal{C}x_k + \mathcal{D}\tilde{u}_k \end{cases} \quad \tilde{u}_k = \begin{pmatrix} u_\Delta \\ u \end{pmatrix} \quad \tilde{y}_k = \begin{pmatrix} y_\Delta \\ y \end{pmatrix} \tag{4.63}$$

Forme exacte

Soit la période d'échantillonnage définie par :

$$h = h_0 + \delta, \qquad \text{avec} \qquad h_{min} - h_0 \leq \delta \leq h_{max} - h_0 \tag{4.64}$$

De l'équation (4.5) on a :

$$\begin{pmatrix} A_d & B_d \\ 0 & I \end{pmatrix} = e^{Mh} = e^{Mh_0}e^{M\delta} = \begin{pmatrix} A_{h_0} & B_{h_0} \\ 0 & I \end{pmatrix} \begin{pmatrix} A_\delta & B_\delta \\ 0 & I \end{pmatrix} \tag{4.65}$$

On en déduit :

$$A_d = A_{h_0}A_\delta \tag{4.66}$$

$$B_d = B_{h_0} + A_{h_0}B_\delta \tag{4.67}$$

On remarque que si $\delta = 0$, alors $A_\delta = I$ et $B_\delta = 0$ et on retrouve bien $A_d = A_{h_0}$ et $B_d = B_{h_0}$. Si l'on définit les incertitudes Δ_1 et Δ_2, par :

$$\begin{aligned} \Delta_1 &= A_\delta - I \\ \Delta_2 &= A_{h_0} B_\delta \end{aligned} \tag{4.68}$$

le transfert du système (4.2) s'exprime alors par :

$$\begin{aligned} G(z) &= D_d + C_d(zI - A_d)^{-1} B_d \\ &= D_d + C_d(zI - A_{h_0}(I + \Delta_1))^{-1}(B_{h_0} + \Delta_2) \end{aligned} \tag{4.69}$$

D'après les expressions (4.69) et (4.70), on peut représenter l'incertitude sur la discrétisation de la matrice A par une incertitude multiplicative inverse et celle sur la discrétisation de B par une incertitude additive. Le schéma bloc de la figure 4.31 représente alors le système $G(z)$.

$$\begin{aligned} (zI - A_d)^{-1} &= [zI - A_{h_0}(I + \Delta_1)]^{-1} \\ &= [(I - A_{h_0}\Delta_1(zI - A_{h_0})^{-1})\ (zI - A_{h_0})]^{-1} \\ &= (I - A_{h_0}\Delta_1(zI - A_{h_0})^{-1})^{-1}\ (zI - A_{h_0})^{-1} \end{aligned} \tag{4.70}$$

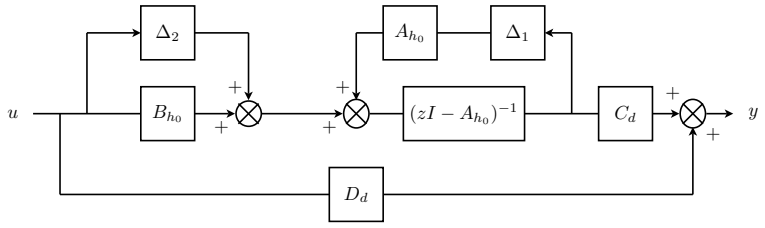


FIG. 4.31 – Représentation incertaine de la discrétisation

Ainsi, la forme LFT de la figure 4.30 est définie par :

$$\begin{aligned} \Delta &= \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \\ \mathcal{A} = A_{h_0} \quad \mathcal{B} &= \begin{pmatrix} A_{h_0} & I & B_{h_0} \end{pmatrix} \\ \mathcal{C} = \begin{pmatrix} I \\ 0 \\ C_d \end{pmatrix} \quad \mathcal{D} &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & I \\ 0 & 0 & D_d \end{pmatrix} \end{aligned} \tag{4.71}$$

Variante Les matrices A_d et B_d dépendent du même paramètre, qui est la période h. Il est donc logique de voir apparaître dans Δ la répétition d'une seule forme d'incertitude. Ainsi d'après l'équation (4.3) et si la matrice A n'est pas singulière, on a :

$$B_\delta = A^{-1}(A_\delta - I)B \tag{4.72}$$

et on peut exprimer Δ_2 par :

$$\begin{aligned}\Delta_2 &= A_{h_0}B_\delta = A_{h_0}A^{-1}(A_\delta - I)B \\ &= A_{h_0}A^{-1}\Delta_1 B \\ &= W_2\Delta_1 W_1\end{aligned} \tag{4.73}$$

La forme LFT de la figure 4.30 est alors définie par :

$$\begin{gathered}\Delta = \begin{pmatrix}\Delta_1 & 0\\ 0 & \Delta_1\end{pmatrix} \\ \mathcal{A} = A_{h_0} \quad \mathcal{B} = \begin{pmatrix}A_{h_0} & W_2 & B_{h_0}\end{pmatrix} \\ \mathcal{C} = \begin{pmatrix}I\\ 0\\ C_d\end{pmatrix} \quad \mathcal{D} = \begin{pmatrix}0 & 0 & 0\\ 0 & 0 & W_1\\ 0 & 0 & D_d\end{pmatrix}\end{gathered} \tag{4.74}$$

Remarque 16 *L'analyse, ou la synthèse d'un correcteur, à partir de la forme $N\Delta$ utilisent la valeur singulière maximale de Δ. Dans notre cas elle est définie par $\bar{\sigma}(\Delta) = \max\{\bar{\sigma}(\Delta_1), \bar{\sigma}(\Delta_2)\}$. La variante de l'équation* (4.74) *permet de réduire le conservatisme si la différence entre $\bar{\sigma}(\Delta_1), \bar{\sigma}(\Delta_2)$ est élevée.*

Forme approchée

L'approche précédente conduit à une incertitude Δ bloc diagonale où chaque bloc est une matrice réelle pleine. Le conservatisme d'une analyse, ou d'une synthèse de correcteur, basée sur un modèle LFT est directement influencé par la structure de l'incertitude Δ et les matrices de mise à l'échelle . Sachant qu'il n'y a qu'un seul paramètre variable, la période δ, la structure de Δ devrait être simple.

On propose ici d'obtenir une matrice de la forme $\Delta = \delta I$. Pour cela on approche les matrices Δ_1 et Δ_2 par des séries de Taylor d'ordre k. Ainsi :

$$\Delta_1 \approx A_\delta - I = \sum_{i=1}^{k} \frac{A^i}{i!}\,\delta^i$$
$$\approx A\delta\ (I + \frac{A\delta}{2}\ (I + \frac{A\delta}{3}\ (I + \dots) \tag{4.75}$$
$$\Delta_2 \approx A_{h_0} B\,\delta = A_{h_0} \sum_{i=1}^{k} \frac{A^i B}{i!}\,\delta^i$$
$$\approx A_{h_0}\ (I + \frac{A\delta}{2}\ (I + \frac{A\delta}{3}\ (I + \dots)B\,\delta \tag{4.76}$$

Les figures 4.32 et 4.33 illustrent l'interconnexion correspondante dans le cas d'une série de Taylor d'ordre 3.

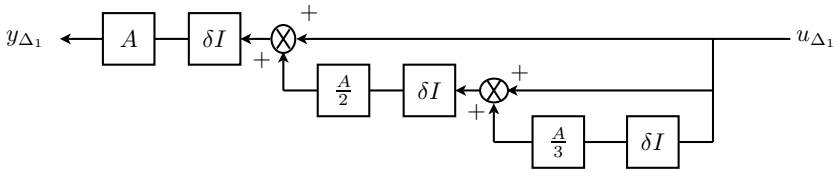


FIG. 4.32 – Approximation de Δ_1 pour $k = 3$

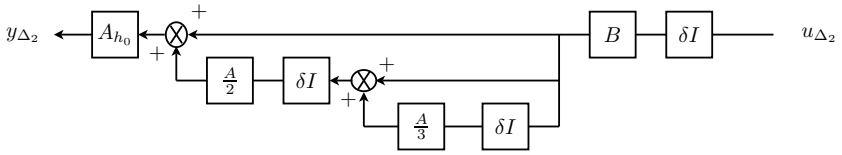


FIG. 4.33 – Approximation de Δ_2 pour $k = 3$

En remplaçant Δ_1 et Δ_2 par leur approximation dans la figure 4.31, la forme LFT de la figure 4.30 est définie par :

$$\Delta = \delta I_{2\times k\times n_s}$$
$$\mathcal{A} = A_{h_0} \quad \mathcal{B} = \begin{pmatrix}\mathcal{B}_1 & \mathcal{B}_2 & B_{h_0}\end{pmatrix} \tag{4.77}$$
$$\mathcal{C} = \begin{pmatrix}\mathcal{C}_1\\ \mathcal{C}_2\\ C_d\end{pmatrix} \quad \mathcal{D} = \begin{pmatrix}\bar{\mathcal{D}} & 0 & 0\\ 0 & \bar{\mathcal{D}} & \mathcal{D}_{2u}\\ 0 & 0 & D_d\end{pmatrix} \tag{4.78}$$

$$\mathcal{B}_1 = \begin{pmatrix} A_a A & 0_{n_s 1} & \dots & \dots 0_{n_s(k-1)} \end{pmatrix} \quad \mathcal{B}_2 = \begin{pmatrix} A_a & 0_{n_s 1} & \dots & \dots & 0_{n_s(k-1)} \end{pmatrix} \tag{4.79}$$

$$\mathcal{C}_1 = \begin{pmatrix} I_{n_s 1} \\ \vdots \\ \vdots \\ I_{n_s k} \end{pmatrix} \quad \mathcal{C}_2 = \begin{pmatrix} 0_{n_s 1} \\ \vdots \\ \vdots \\ 0_{n_s k} \end{pmatrix} \quad \bar{\mathcal{D}} = \begin{pmatrix} 0_{n_s} & \frac{A}{2} & \dots & 0_{n_s} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{A}{k} \\ 0_{n_s} & \dots & \dots & 0_{n_s} \end{pmatrix} \quad \mathcal{D}_{2u} = \begin{pmatrix} B \\ \vdots \\ \vdots \\ B \end{pmatrix} \tag{4.80}$$

Cette formulation permet de simplifier la structure de l'incertitude sous la forme $\Delta = \delta I_{2*k*n_s}$. Néanmoins, la taille du bloc incertitude est plus grande que celle de l'approche exacte car elle est multipliée par k. Il y a donc un compromis entre la précision de l'approximation et la complexité de la synthèse, similaire à celui de la modélisation LPV polytopique.

4.5.3 Synthèse du correcteur

Une fois le modèle LFT de la discrétisation paramétrée obtenu, il est facile de construire un système augmenté d'une forme similaire à la figure 4.34, ceci grâce aux propriétés de la LFT. Ce système augmenté contient alors le procédé discrétisé, son interconnexion avec le contrôleur et les entrées/sorties exogènes et enfin les pondérations fréquentielles spécifiant les objectifs en boucle fermée.

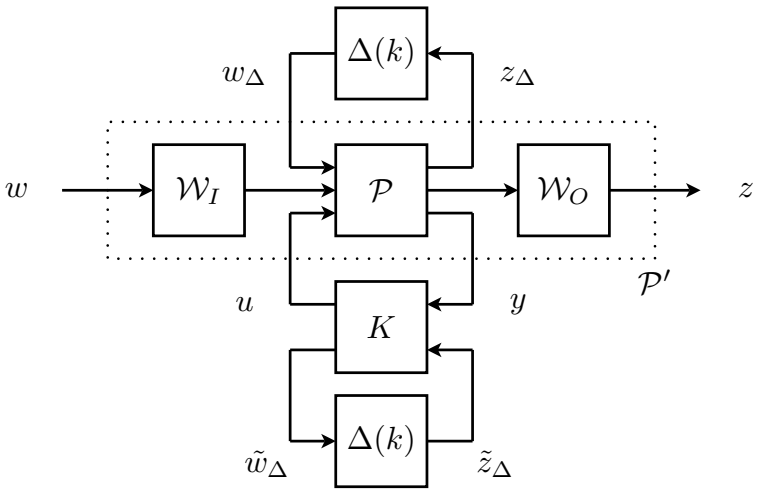


FIG. 4.34 – Problème standard étendu

On reconnaît ici une formulation similaire à celle de la figure 4.28 à partir de laquelle on peut appliquer le théorème 3.

Cette méthodologie a été appliquée sur l'exemple du paragraphe 4.4. Les deux formes, exactes et approchées, de la discrétisation paramétrée ont été testées. Malheureusement les

résultats numériques sont pour le moment aberrants. Nous nous limiterons donc à l'exposé de la formulation sans pouvoir démontrer son intérêt sur un exemple illustratif.

Conclusion du chapitre

Dans un premier temps, nous avons formulé la discrétisation paramétrée d'un modèle linéaire à temps continu. Le modèle obtenu représente l'ensemble des discrétisations pour un intervalle de périodes d'échantillonnage connu et borné. Deux formulations ont été proposées, motivées par l'objectif d'obtenir un modèle linéaire à paramètres variants. La première repose sur l'usage d'une série de Taylor et d'un modèle LPV sous forme polytopique. Deux constructions du polytope sont présentées dont l'une permet de réduire la dimension du polytope sans perdre d'information sur le modèle. La seconde formulation repose sur l'usage de la transformée linéaire fractionnelle. Deux constructions sont là aussi proposées. La première est exacte mais conduit à un bloc d'incertitude non structuré, la seconde, via l'utilisation d'une approximation de Taylor, simplifie le bloc d'incertitude en le structurant.

Dans un deuxième temps, nous avons utilisé les outils de contrôle des systèmes LPV pour construire un correcteur à paramètres variants. Cette approche s'est montrée concluante pour la première formulation. L'étude détaillée d'un exemple académique illustre la faisabilité et l'avantage d'une commande à période variable doit le comportement en fonction de la période est prévisible. Des résultats numériques difficiles à interpréter ne permettent pas, pour le moment, de conclure sur la seconde formulation.

Il faut maintenant vérifier, par une expérimentation, que l'approche reposant sur une modèle LPV polytopique est utilisable sur un procédé réel. Nous verrons pour cela, dans le prochain chapitre, son application au contrôle d'un pendule instable de la forme d'un "T".

Chapitre 5

Application d'un correcteur à période variable au contrôle d'un pendule T

Ce chapitre est consacré à l'application sur un procédé expérimental de la commande à période d'échantillonnage variable présentée au chapitre précédent. L'objectif est ici de montrer la faisabilité de l'approche et non le contrôle parfait d'un pendule T. Ce procédé a été choisi pour sa disponibilité et sa complexité.

Dans un premier temps nous détaillons le procédé expérimental et sa modélisation. Dans un deuxième temps nous expliquons la formulation de la synthèse avec les outils présentés au chapitre précédent. Enfin le correcteur à période variable obtenu est testé en simulation puis sur le procédé physique.

5.1 Description de l'expérience

5.1.1 Présentation

Le procédé de la photo 5.1 est un pendule inversé en forme de "T". Il est développé par la société Educational Control Products, sous le modèle 505.

Le pendule, schématisé par la figure 5.2, est composé de deux barres. L'une, verticale, est en rotation autour d'un axe relié au support. L'autre, horizontale, coulisse perpendiculairement à la première dans un guide situé à son sommet. Des masses optionnelles permettent de changer la dynamique du procédé. Ainsi, deux masselottes démontables sont fixées aux extrémités de la barre horizontale pour en modifier l'inertie. Une autre masselotte, fixée à une distance variable de la base de la barre verticale, permet d'ajuster la position du centre de masses.

Le pendule est actionné par l'application d'une force u sur la barre horizontale pour la

FIG. 5.1 – Photo du pendule T

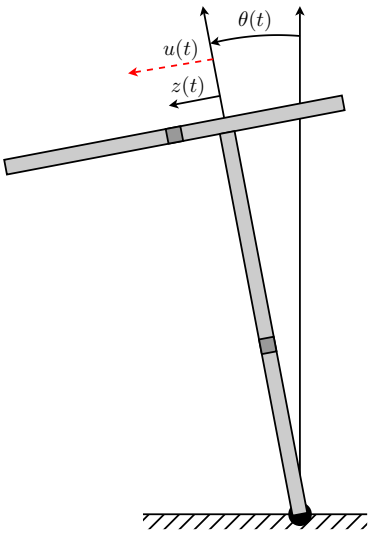


FIG. 5.2 – Schéma du pendule T

translater. Pour cela, une crémaillère, solidaire de la barre horizontale est mis en mouvement par un moteur à courant continu.

L'angle θ, positif dans le sens trigonométrique, est mesuré par un capteur relié à l'axe de rotation. La déplacement z de la barre horizontale par rapport à sa position médiane, positif vers la gauche, est mesuré par un capteur relié à l'axe du moteur.

Le moteur est commandé en couple par un asservissement local du courant. Cette boucle locale, dont la dynamique est négligeable devant celle du procédé mécanique, est assimilée à un gain statique unitaire. La dynamique des capteurs est suffisamment rapide pour être négligée et modélisée par un gain unitaire.

5.1.2 Modélisation

Une modélisation mécanique du pendule, sans tenir compte des frottements, conduit au modèle non linéaire suivant :

$$\begin{pmatrix} m_1 & m_1 l_0 \\ m_1 l_0 & \bar{J} \end{pmatrix} \begin{pmatrix} \ddot{z} \\ \ddot{\theta} \end{pmatrix} + \begin{pmatrix} 0 & -m_1 z\dot{\theta} \\ 2m_1 z\dot{\theta} & 0 \end{pmatrix} \begin{pmatrix} \dot{z} \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} -m_1 \sin\theta \\ -(m_1 l_0 + m_2 l_c)\sin\theta - m_1 z\cos\theta \end{pmatrix} g = \begin{pmatrix} u \\ 0 \end{pmatrix} \quad (5.1)$$

où la dépendance temporelle des variables d'état est implicite et les valeurs des paramètres sont listées dans le tableau 5.1.

Nom	Valeur	Description
m_1	0.217 kg	Masse de la barre horizontale (avec les masselottes)
m_2	1.795 kg	Masse de la barre verticale (avec la masselotte)
l_0	0.33	Longueur de la barre verticale
l_c	-0.032 m	Position du centre de gravité de la barre verticale
g	9.81 m.s^{-2}	Accélération gravitationnelle
$\bar{J}$	0.061 Nm^2	Moment d'inertie nominal

TAB. 5.1 – Paramètres du modèles

En définissant le vecteur d'état $x = [z, \dot{z}, \theta, \dot{\theta}]$, l'équation (5.3) se ré-écrit :

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -l_0\dot{x}_4 + x_1x_4^2 + g\sin x_3 + \dfrac{u}{m_1} \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \dfrac{1}{J_0(x_1) - m_1l_0^2}\left(+g(m_1x_1\cos x_3 + m_2l_c\sin x_3)\right. \\ \qquad \left. -m_1(l_0x_4 + 2x_2)x_1x_4 - l_0u\right) \end{cases} \tag{5.2}$$

$$\text{avec} \quad J_0(x_1) = \bar{J} + m_1x_1^2 \tag{5.3}$$

La linéarisation autour du point d'équilibre $x = [0, 0, 0, 0]$ donne la représentation :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{5.4}$$

avec :

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{-l_0gm_1}{\bar{J}-m_1l_0^2} & 0 & \frac{-l_0gm_2l_c}{\bar{J}-m_1l_0^2} + g & 0 \\ 0 & 0 & 0 & 1 \\ \frac{gm_1}{\bar{J}-m_1l_0} & 0 & \frac{gm_2l_c}{\bar{J}-m_1l_0^2} & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ \frac{l_0^2}{\bar{J}-m_1l_0^2} + \frac{1}{m_1} \\ 0 \\ \frac{-l_0}{\bar{J}-m_1l_0^2} \end{pmatrix} \tag{5.5}$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

L'application numérique avec les valeurs du tableau 5.1 produit les matrices suivantes :

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -18.79 & 0 & 14.82 & 0 \\ 0 & 0 & 0 & 1 \\ 56.92 & 0 & -15.18 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 7.52 \\ 0 \\ -8.82 \end{pmatrix} \tag{5.6}$$

Les pôles du modèle linéaire sont $p_{1,2} = \pm 6,788i$, $p_3 = -3.481$ et $p_4 = 3.481$. Le système est donc instable.

5.1.3 Adaptation du modèle

Le précédent modèle ne tient pas compte des frottements présents sur le procédé expérimental. Nous avons constaté que des frottements affectent la translation de la barre horizontale et génèrent des oscillations. Pour avoir un modèle plus réaliste nous ajoutons un terme de frottements visqueux sur la vitesse de déplacement de la barre horizontale. Il est fort probable que des frottements secs aient aussi un impact mais il est impossible de les modéliser sous une forme linéaire, c'est pourquoi nous les négligeons.

Le modèle non linéaire de l'équation (5.3) est modifié tel que :

$$\dot{x}_2 = -l_0\dot{x}_4 + x_1x_4^2 + g\sin x_3 + \frac{u}{m_1} - \frac{f_{v_z}}{m_1}x_2 \tag{5.7}$$

La valeur de f_{v_z} a été choisie empiriquement afin d'obtenir en simulation un comportement similaire à l'expérimentation. Après linearisation et l'application numérique avec $f_{v_z} = 0,1$, la matrice A est définie par l'équation (5.8). Les pôles du système deviennent $p_{1,2} = -0,122 \pm 6,784i$, $p_3 = -3,592$ et $p_4 = 3,376$. La paire de pôles imaginaires purs est remplacée par une paire de pôles complexes conjugués dont la partie réelle est négative. Ceci est plus réaliste car les oscillations du procédé sont forcément amorties et une paire de pôles imaginaires purs représente un système oscillant non amorti.

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -18.79 & -0.46 & 14.82 & 0 \\ 0 & 0 & 0 & 1 \\ 56.92 & 0 & -15.18 & 0 \end{pmatrix} \tag{5.8}$$

La figure 5.3 présente le tracé de la valeur singulière maximale du modèle adapté. Rappelons que ce tracé indique le gain dans le pire cas entre le vecteur d'entré et celui de sortie. On retiendra principalement un pic de résonance de 18 dB à la pulsation 6,78 rad/s. Le système étant instable, c'est dans cette gamme de fréquences que le système sera le plus difficile à contrôler.

Le contrôle du procédé est apparu difficile avec pour seules mesures les positions z et θ. Pour bénéficier de la mesure de l'état complet nous estimons les vitesses $\dot{z}$ et $\dot{\theta}$ via l'usage d'un filtre se comportant comme une dérivé dans la bande passante utile pour ce procédé. Par la suite nous considérons la matrice C égale à l'identité.

Les fonctions de transfert reliant l'entrée u aux sorties z et θ présentent toutes les deux un zéro instable. Dans les cas où l'on mesure soit les deux positions, soit l'état complet,

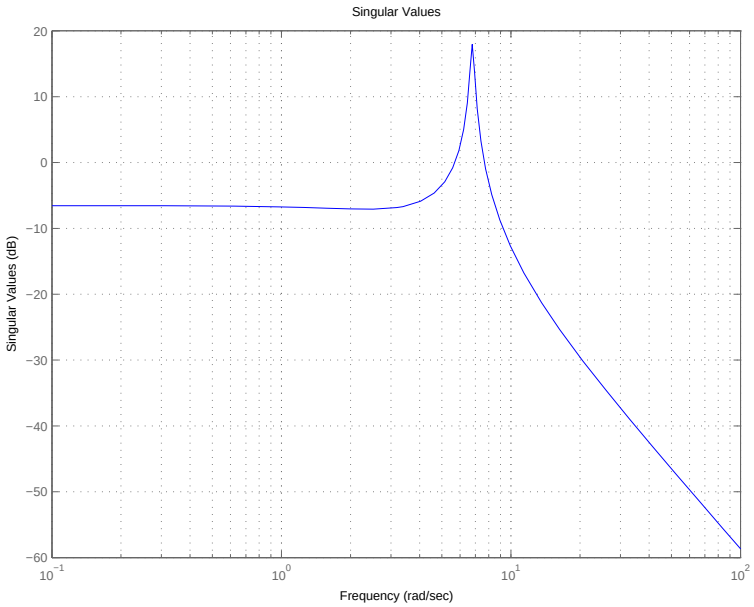


FIG. 5.3 – Tracé de la valeur singulière maximale du modèle à temps continu

ce système ne possède pas de zéros de transmission. Néanmoins il faut s'attendre à un comportement à non minimum de phase, ce que nous constaterons par la suite.

5.2 Conception du correcteur

5.2.1 Objectifs en boucle fermée

Différents travaux, (Witrant, 2005; Natale et al., 2004), réalisés au laboratoire sur cette manipulation ont montré que le contrôle de ce procédé est délicat. Dans ce contexte notre objectif en boucle fermée sera avant tout de stabiliser le système et de l'accélérer si possible.

La période d'échantillonnage est comprise entre 1 et 3 ms. Bien que ces valeurs soient bien trop faibles au regard des pulsations du modèle linéaire et du critère (1.1), les précédentes expérimentations avec une loi de commande issue d'une synthèse H_∞ à temps continu ont nécessité une période d'échantillonnage comprise entre 1 à 5 ms. N'oublions pas que ce critère empirique est valable pour un système linéaire or le procédé est ici non linéaire et la présence de frottements complique son contrôle. Il est donc logique de devoir réduire la période d'échantillonnage.

Les objectifs de la synthèse sont formulés avec le problème standard à temps discret de l'équation (5.9) illustré par la figure 5.4.

$$\left\| \begin{matrix} W_e(I - MS_yGK_1) & W_eMS_yG \\ W_uS_uK_1 & W_uT_u \end{matrix} \right\|_\infty \leq \gamma \tag{5.9}$$

avec :

$$\begin{aligned} K &= \begin{bmatrix} K_1(z) & K_2(z) \end{bmatrix} & M &= \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \\ S_u &= (I - K_2(z)G(z))^{-1} & S_y &= (I - G(z)K_2(z))^{-1} \\ T_u &= -K_2(z)G(z)(I - K_2(z)G(z))^{-1} \end{aligned} \tag{5.10}$$

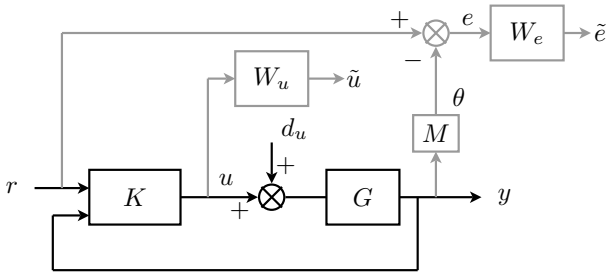


FIG. 5.4 – Problème standard étudié

Les objectifs en boucle fermée sont définis par les gabarits fréquentiels à temps continu de l'expression (5.12), dont la pulsation est fonction de la période d'échantillonnage :

$$W_e(p, f) = \frac{p\, M_S + \omega_S(f)}{p + \omega_S\, \epsilon_S} \qquad \omega_S(f) = h_{min}\, \omega_{S_{max}}\, f \tag{5.11}$$

$$W_u(p, f) = \frac{1}{M_U} \tag{5.12}$$

où $\omega_{S_{max}} = 1{,}5$ rad/s, $M_S = 2$, $\epsilon_S = 0{,}01$ et $M_U = 5$, $f \in [0.33; 1.00]\ kHz$.

5.2.2 Discrétisation du modèle

L'obtention d'un modèle LPV polytopique à temps discret passe par la discrétisation paramétrée du modèle. Les figures 5.5 et 5.6 montrent l'erreur de discrétisation selon le critère (4.17). L'erreur varie entre 0 et 2,6. On choisira k=2 qui implique un polytope de dimension raisonnable et une erreur inférieure à -40 dB (figure 5.6). En réduisant le polytope, le modèle polytopique est alors composé de 3 sommets.

Après la discrétisation des gabarits fréquentiels, la construction du système augmenté et le filtrage de l'entrée de commande par un filtre passe-bas du premier ordre de pôle égal à zéro, on obtient un système augmenté à temps discret du 6ème ordre.

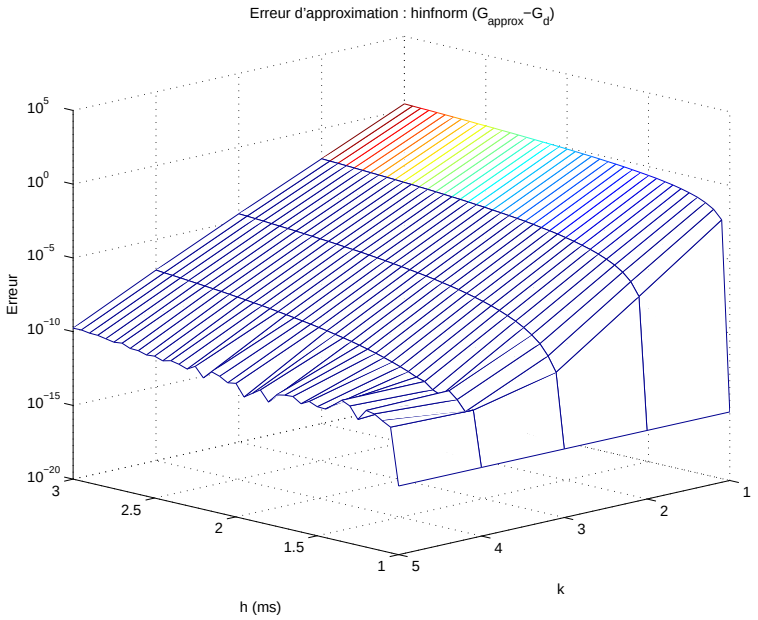


FIG. 5.5 – Erreur d'approximation

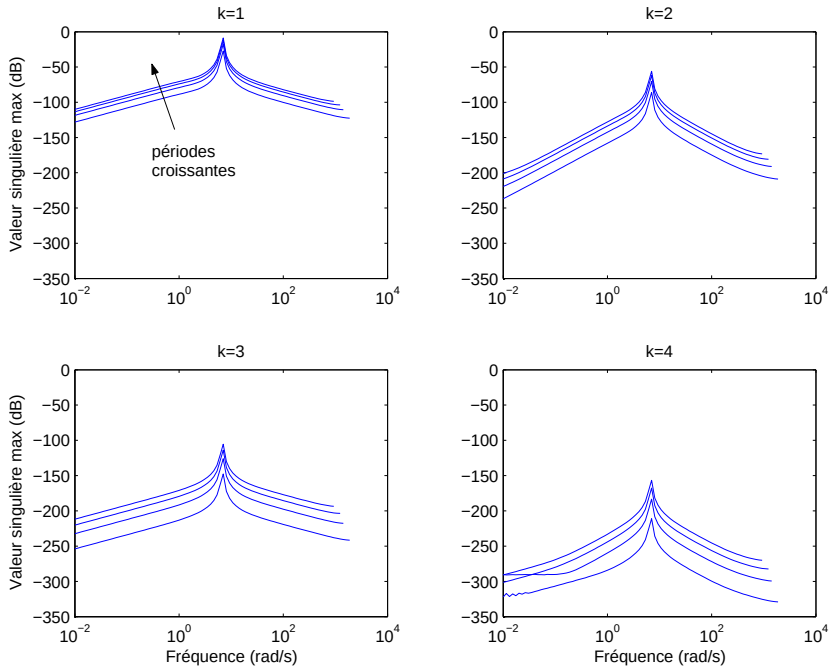


FIG. 5.6 – Répartition fréquentielle de l'erreur d'approximation

5.2.3 Synthèse du correcteur

Nous utilisons la méthode de synthèse décrite dans le théorème 1, c'est à dire la synthèse d'un correcteur LPV polytopique où la stabilité est démontrée par l'usage d'une fonction de Lyapunov constante. A l'issue de la synthèse nous obtenons $\gamma = 1{,}13$.

La synthèse du théorème 2 a été également testée. Malgré l'usage d'une fonction de Lyapunov dépendante du paramètre variant, cette approche n'apporte ici aucun bénéfice qui aurait pu se manifester par une valeur de γ plus petite. Pour la suite de l'expérimentation, nous conservons la première méthode.

La figure 5.7 présente les fonctions de sensibilité obtenues. La sensibilité S_e montre une erreur statique inférieure à 0,01, sa bande passante varie de 0,4 à 1,2 rad/s, c'est à dire dans un rapport de 3, la vitesse de l'asservissement devrait donc varier dans le même rapport. Le pic de $S_u K_1$ varie de 1,2 à 10,8 dB ce qui est une valeur raisonnable pour le gain u/r. Néanmoins, cette sensibilité est probablement trop forte en hautes fréquences. La sensibilité MS_yG montre une atténuation à toutes les fréquences de l'effet d'une perturbation en entrée sur l'erreur d'asservissement, néanmoins sa plus faible valeur (-22 dB) est obtenue près de la fréquence naturelle du procédé. Enfin la sensibilité MS_yGK_1, qui représente la boucle fermée, a une bande passante variant de 0,7 à 7 rad/s.

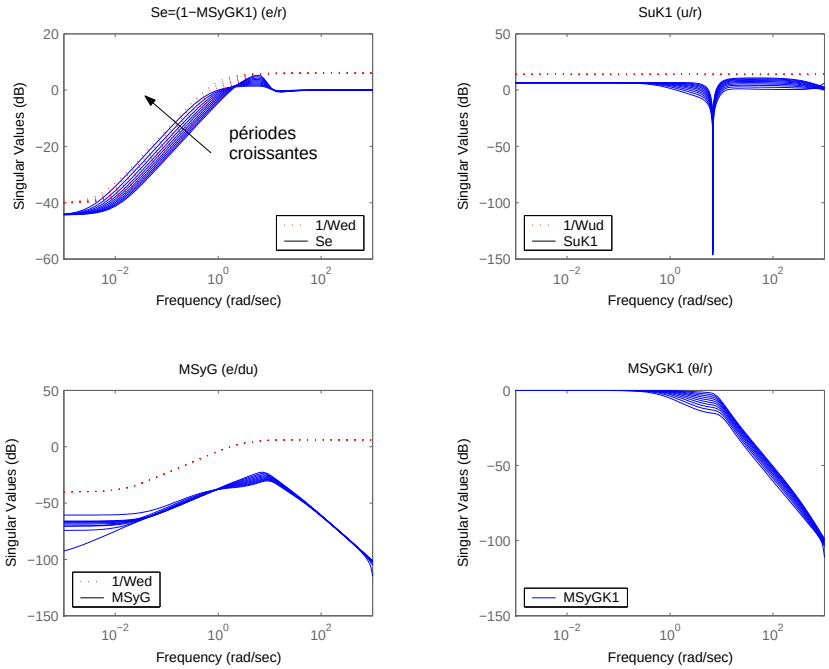


FIG. 5.7 – Fonctions de sensibilité

5.2.4 Simulations

La figure 5.8 présente la réponse temporelle de l'interconnexion entre le correcteur à temps discret et le modèle non linéaire du procédé à temps continu. Dix tracés sont superposés pour dix valeurs de la période d'échantillonnage comprises entre 1 et 3 ms.

L'état initial du pendule n'est pas l'équilibre mais une position réaliste d'initialisation de l'expérimentation. L'objectif de cette valeur initiale est de vérifier que le pendule peut s'équilibrer à l'initialisation sans dépasser la saturation de l'actionneur, c'est à dire la limite des ±10 cm de débattement de la barre horizontale.

Le temps de réponse varie de 1,1 à 4,8 s, soit dans un rapport de 4,3. Il n'a pas de dépassement, ce qui est conforme à la sensibilité SyGK1. Par contre on voit apparaître le comportement d'un système à non minimum de phase.

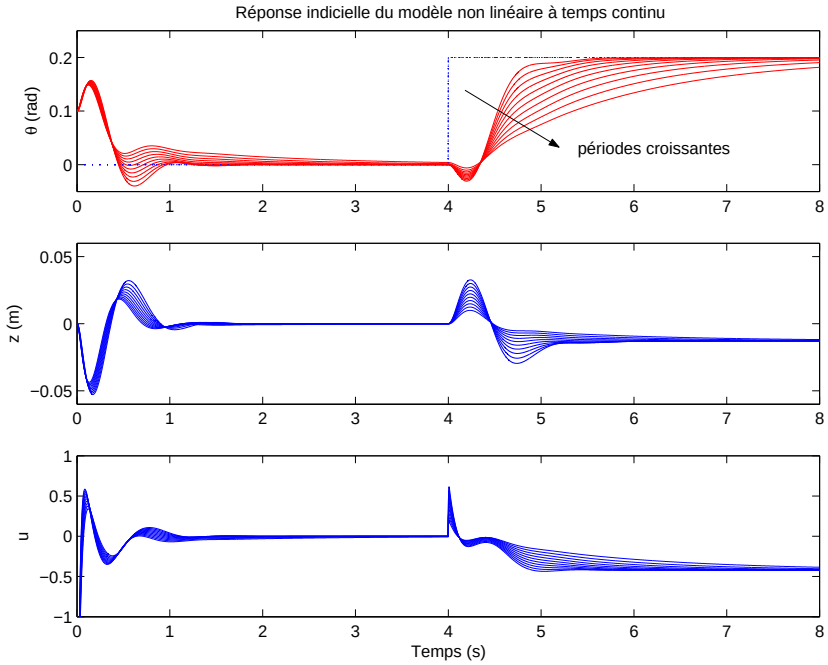


FIG. 5.8 – Réponse temporelle du procédé non linéaire à temps continu

Nous appliquons à présent le correcteur obtenu sur un scénario où la période varie en ligne entre 1 et 3 ms selon une sinusoïde de pulsation 0,15 rad/s. Deux simulations sont présentées sur les figures 5.9 et 5.10.

Conformément aux observations précédentes, le temps de réponse du pendule est adapté à la valeur de la période d'échantillonnage. Que ce soit en réponse à une variation douce de la période, comme celle d'un sinus, ou d'une variation plus brutale, comme celle d'un créneau, le pendule n'est pas perturbé par la variation de la période. Il en est de même pour la commande. Notons que les pics présents sur la commande sont la conséquence d'un

changement brutal de la référence r et non de la période. Ces pics ont une amplitude plus élevée quand la période est la plus petite, c'est à dire quand la bande passante du système en boucle fermée est la plus élevée. Ce comportement est donc tout à fait logique.

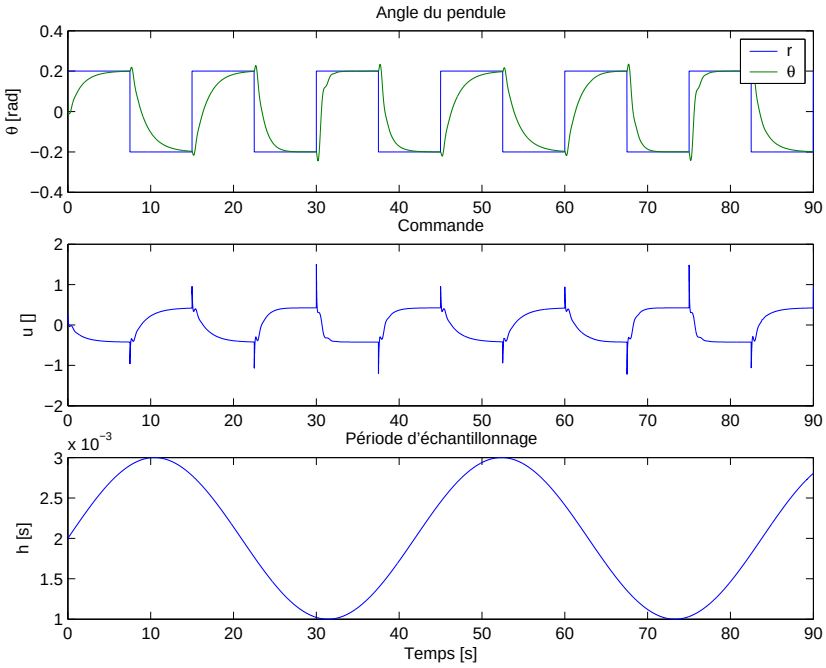


FIG. 5.9 – Simulation du pendule T en réponse à une période sinusoïdale

5.2.5 Expérimentations

Nous appliquons maintenant les mêmes scénarios sur le procédé réel. L'expérimentation a été réalisée sur le procédé de la photo 5.1 via l'utilisation des logiciels Matlab et Simulink et de leurs extensions Real-time Workshop et xPC Target. Les résultats obtenus sont illustrés sur les figures 5.11 et 5.12.

Fidèle à la simulation, le temps de réponse du pendule s'adapte à la valeur de la période. Ainsi quand la période est maximale, le temps de réponse est minimale. De même la commande et la réponse du pendule ne semblent pas être perturbées par la variation de la période, y compris dans le cas ou cette dernière change brusquement entre 1 et 3 ms (voir figure 5.12).

La principale différence entre la simulation et l'expérimentation vient de la commande. Cette dernière est en pratique beaucoup plus bruitée qu'en simulation. Ceci est la conséquence du gabarit W_u qui est constant sur toute la plage fréquentielle. Ceci montre une faiblesse de l'approche puisqu'il est difficile lors de la synthèse de contraindre la sensibilité de la commande aux bruits en hautes fréquences.

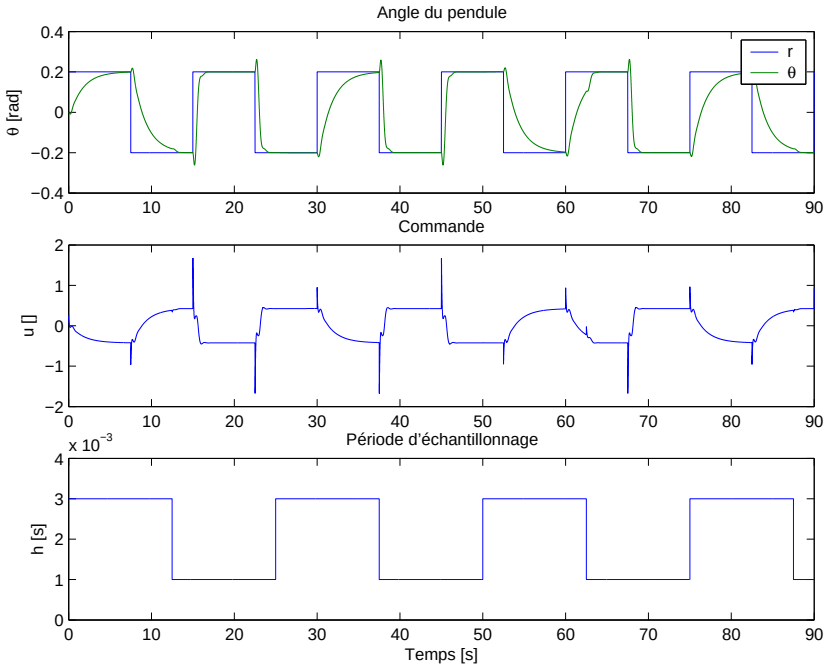


FIG. 5.10 – Simulation du pendule T en réponse à une période en créneaux

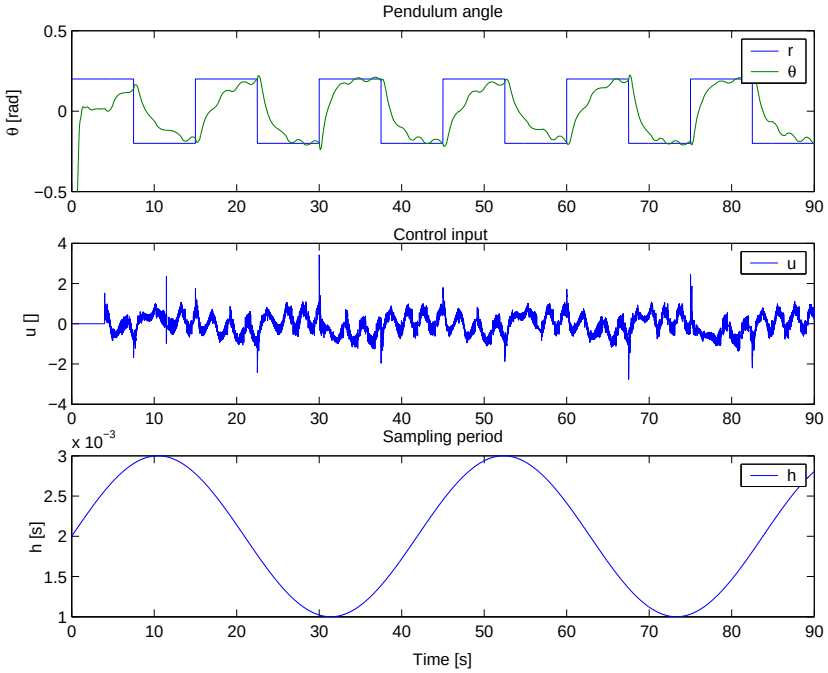


FIG. 5.11 – Pendule T expérimental en réponse à une période sinusoïdale

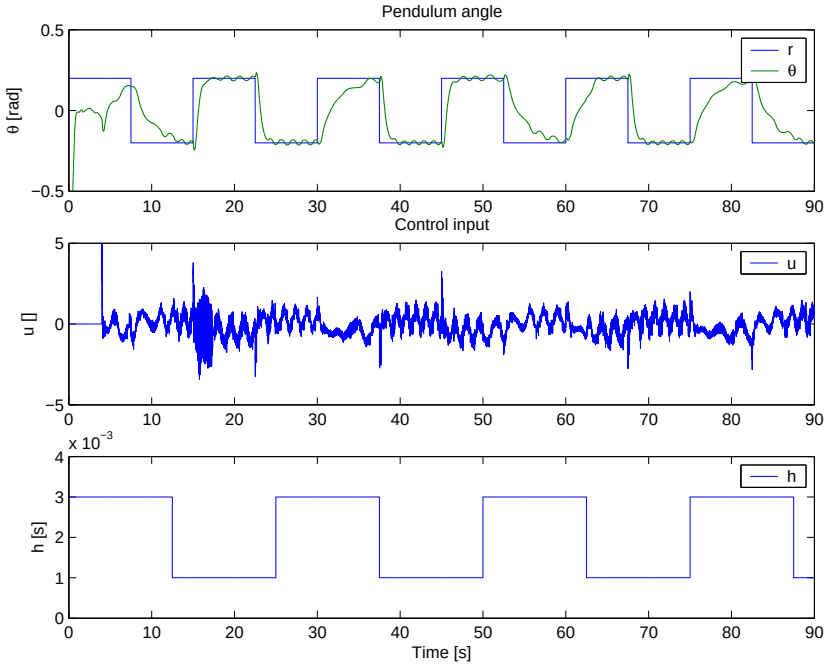


FIG. 5.12 – Pendule T expérimental en réponse à une période en créneaux

Une autre différence concerne les oscillations résiduelles qui apparaissent sur l'angle et sur la commande. Comme énoncé lors de la modélisation, le procédé possède des frottements au niveau de la translation de la barre horizontale. Un terme de frottement visqueux a été introduit empiriquement dans le modèle pour tenter de les atténuer mais il nous est impossible de prendre en compte les frottements secs dans ce correcteur linéaire. Des essais ont montré que l'atténuation de ces oscillations est liée à la sensibilité du correcteur en hautes fréquences. Le bruit présent sur la commande, par manque d'atténuation en hautes fréquences de la part du correcteur, a un effet bénéfique sur les frottements. En effet le moteur vibre légèrement et réduit l'effet des frottements secs. Cet artifice bien que bénéfique sur cette expérimentation n'est pas forcément une solution viable pour la durée de vie du moteur. Il faudrait d'une part contrôler la fréquence et l'amplitude des bruits présents sur la commande et d'autre part vérifier qu'ils ne sont pas destructifs pour le moteur.

Conclusion du chapitre

Dans ce chapitre nous avons appliqué une synthèse de correcteurs à période d'échantillonnage variable présentée au chapitre 4 sur un procédé expérimental du laboratoire.

Il apparaît tout d'abord que la seconde synthèse proposée dans le précédent chapitre n'apporte aucun bénéfice par rapport à la première. Alors que l'usage d'une fonction de Lyapunov devrait réduire le conservatisme de la première approche, nous ne le constatons

pas sur notre exemple.

Le procédé expérimental, plus complexe que l'exemple académique du chapitre 4, a permis de prouver la faisabilité concrète de l'approche. Il a été développé pour regrouper en une application plusieurs difficultés courantes de la commande des systèmes dynamiques (non linéarité, instabilité, contrainte d'actionneur). Obtenir une commande qui fonctionne sur cette expérience est donc un point positif.

La simulation et l'expérimentation ont montré un comportement prévisible du procédé en boucle fermée en réponse aux variations de la période. On garantit ainsi que la commande sera plus performante quand la période est plus petite et ce, de façon progressive. Les changements de période, même rapides, ne produisent pas de comportements brusques sur la commande. C'est ici tout l'intérêt d'utiliser une commande LPV par rapport à une commande à gains séquencés ou à changement de modes.

Enfin on notera la difficulté à contraindre le comportement du correcteur en hautes fréquences. Cette sensibilité aux bruits a permis ici d'atténuer l'effet des frottements secs mais au détriment de vibrations qui peuvent être néfastes pour l'actionneur. Ce problème devra être approfondi dans le futur.

Chapitre 6

Vers l'interaction commande/ordonnancement

Dans ce chapitre nous exposons deux structures d'interaction entre la commande et l'ordonnancement. Dans un premier temps, la régulation d'ordonnancement et les lois de commande à période variable sont regroupées dans une application commune. Dans un deuxième temps, un ordonnancement adapté en ligne à la qualité de la commande est exposé sous la forme d'un exemple.

6.1 Ordonnancement régulé de tâches de commande

Dans le chapitre 2 la régulation de la charge processeur d'un ensemble de tâches périodiques a été abordée. Pour cela un régulateur d'ordonnancement mesure le temps d'exécution et adapte la période des tâches. Si ces dernières sont des lois de commande, il est nécessaire de mettre à jour le correcteur en fonction de la période, ce qui a motivé le développement d'une commande à période variable. Nous regroupons ici ces deux outils dans un exemple d'application.

L'intérêt est d'adapter l'ordonnancement aux variations de l'usage du processeur tout en maîtrisant les conséquences sur les commandes. Les causes des fluctuations des ressources sont :

- le départ ou l'arrivé d'une tâche : dans des systèmes complexes, toutes les tâches ne sont pas forcément périodiques.
- la variation du temps d'exécution d'une tâche : par exemple des optimisations en ligne ou du traitement d'images.
- la variation de la capacité de calcul : par économie d'énergie, certains processeurs adaptent leur fréquence de fonctionnement et calculent plus ou moins vite.

Notons que la situation est similaire lors du partage d'un réseau de communication. Il

existe un parallèle entre la charge processeur et la bande passante d'un réseau.

6.1.1 Description de l'exemple

Notre exemple consiste à contrôler deux pendules stables, identiques à l'exemple académique du chapitre 4 et dont on utilise le correcteur. Rappelons que ce dernier est obtenu pour une série de Taylor d'ordre 2, que son polytope, de taille réduite, comporte 3 sommets et que sa période varie de 40 à 120 ms.

Chaque contrôleur est implémenté comme une tâche périodique sur un système temps réel muni d'un ordonnancement préemptif à priorités fixes. Le système comporte deux autres tâches : le régulateur d'ordonnancement et une tâche perturbatrice. Le tableau 6.1 regroupe les caractéristiques des tâches où la priorité la plus forte vaut 1.

Tâche	Priorité	Période (ms)	Temps d'exécution (ms)
reg	1	500	10
pend1	2	40 à 120	20
perturb	3	10	2
pend2	4	40 à 120	20 à 24

TAB. 6.1 – Caractéristiques des tâches

Sur les quatre tâches, seules les tâches de commande ont une période modifiable. Le régulateur d'ordonnancement mesure le temps d'exécution de *pend1*, *pend2*, *perturb* et agit sur la période des deux commandes. Sa structure est rappelée sur la figure 6.6 où $K(z)$ est le régulateur d'ordonnancement.

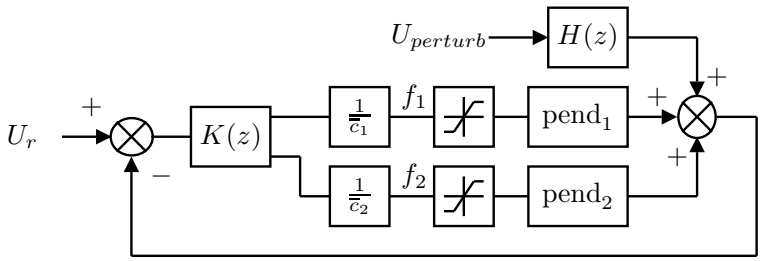


FIG. 6.1 – Schéma bloc de l'implémentation du régulateur

Le régulateur $K(z)$ est obtenu par la méthode présentée dans le chapitre 2 avec les paramètres suivants : $\lambda_1 = 0,3$, $\lambda_2 = 0,3$, $M_s = 2$, $\omega_s = 1$ rad/s et $\epsilon = 0,05$. La répartition de charge est spécifiée par $M = [-0.5\ 1]$ et $W_x = 100$ pour que la part allouée à *pend1* soit le double de celle allouée à *pend2*.

Le système est simulé avec la boîte à outils TrueTime (Henriksson et al., 2002) disponible pour Matlab. Le scénario de simulation est le suivant : les deux pendules sont asservis sur

un signal carré de période 6s, de rapport cyclique 50% et d'amplitude 1. La régulation de charge débute avec une référence à 80 % puis diminue à 60 % à $t = 5s$. Le temps d'exécution de la tâche *pend2* est initialement 20 ms puis augmente à 24 ms à $t = 9s$. Enfin la tâche perturbatrice apparaît à $t = 12s$.

6.1.2 Résultats

La figure 6.2 présente l'évolution de l'ordonnancement. On peut estimer le temps de réponse de la régulation d'ordonnancement, en réponse à une variation de sa référence, à environ 4s. Lors des 10 premières secondes aucune saturation des périodes n'est atteinte. Sachant que *pend1* et *pend2* ont le même temps d'exécution, on retrouve le rapport de répartition de charge dans les périodes, c'est à dire que la période de *pend2* est 2 fois supérieure à celle de *pend1* ce qui correspond à 2 fois plus de ressources pour *pend1* que *pend2*. A $t = 9s$, la variation du temps d'exécution de *pend2* est très rapidement corrigée puisque la dynamique de la régulation de l'ordonnancement n'intervient pas (voir chapitre 2). A $t = 12s$, l'arrivée de la tâche perturbatrice, qui nécessite 20% des ressources, crée un pic de charge qui est rejeté par la régulation de l'ordonnancement. Notons enfin qu'à partir de $t = 10s$, la période de pend2 étant à son maximum, la répartition de charge entre les deux tâches de commande n'est plus respectée.

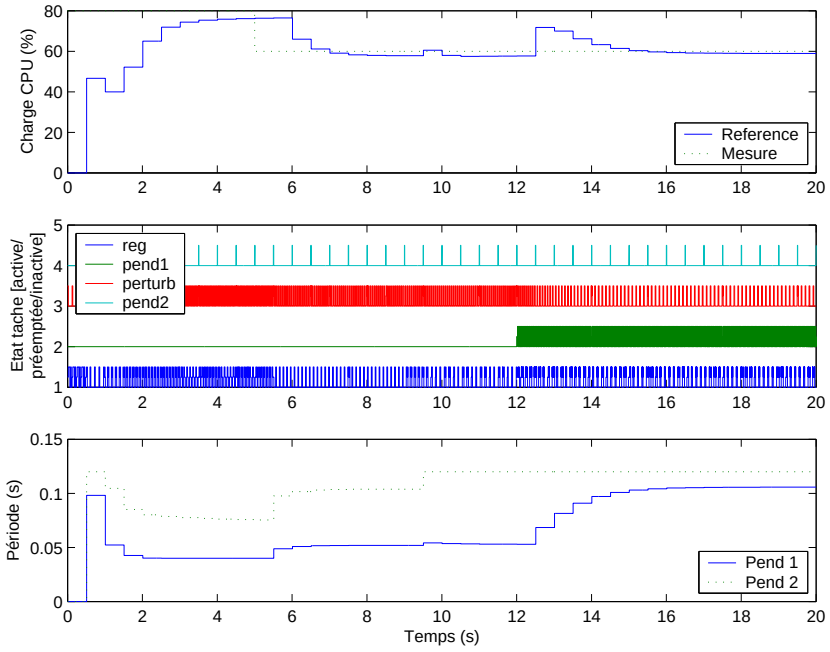


FIG. 6.2 – Evolution de l'ordonnancement

Ce scénario illustre la flexibilité apportée par la régulation de l'ordonnancement. Dans

le cas d'un ordonnancement non adapté, il aurait fallu dimensionner ce dernier en tenant compte de l'augmentation du temps d'exécution de pend2 et de la présence de la tâche perturbatrice ce qui aurait conduit pendant le début de la simulation à une sous utilisation du calculateur.

Cette flexibilité n'est possible que si la période des tâches de commande peut être modifiée en ligne, ce qui est notre cas. Le comportement des pendules est présenté sur le graphique 6.3.

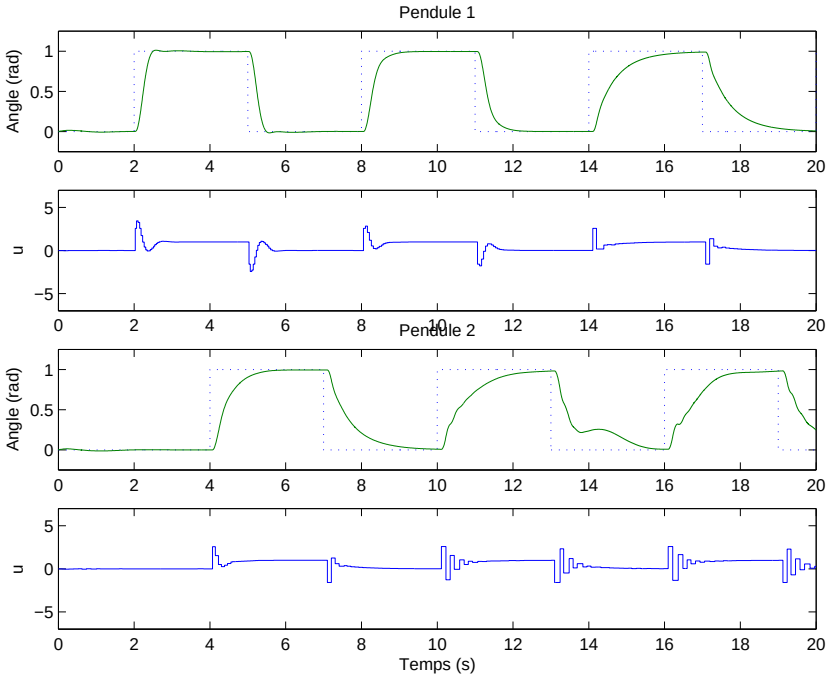


FIG. 6.3 – Evolution des pendules (objectifs variables)

Comme on pouvait s'y attendre, la rapidité du système évolue en fonction de la période de la commande. On note cependant que la réponse du second pendule est perturbée vers $t = 13s$. Cela correspond à la période transitoire de la régulation de l'ordonnancement. La tâche *pend2* étant la moins prioritaire, elle subit plus de préemptions ce qui induit une latence entrée/sortie plus importante et donc une perturbation de la commande. Une fois la charge revenue à 60% les préemptions sont moins nombreuses et la réponse du pendule est moins perturbée.

Cette perturbation sur le second pendule soulève deux problèmes. D'une part, l'ordonnancement étant à priorités fixes avec des priorités choisies arbitrairement, il est difficile d'utiliser 100% du processeur et d'avoir peu de perturbations temporelles. Le choix de la référence de charge est un compromis entre l'usage du processeur et la perturbation des commandes. D'autre part la régulation de l'ordonnancement ne tient pas compte de la qua-

lité de la commande. Or pour une charge processeur fixe, les perturbations induites par l'ordonnancement sur la commande varient selon le nombre de tâches et leurs caractéristiques (priorité, période, temps d'exécution). La seule mesure de la charge processeur est donc insuffisante pour estimer la qualité de la commande. Néanmoins la tendance est bonne car plus la charge est élevée et plus les perturbations sont probables.

Notes sur les objectifs de commande constants

Dans le chapitre 4 deux correcteurs à période variable ont été comparés, l'un avec des objectifs en boucle fermée variables, l'autre avec des objectifs constants.

Nous reprenons ici le scénario du paragraphe précédent où les commandes des deux pendules sont maintenant à objectifs constants. L'évolution de l'ordonnancement est identique à la figure 6.2 car elle est indépendante des commandes. La figure 6.4 présente l'évolution des deux pendules.

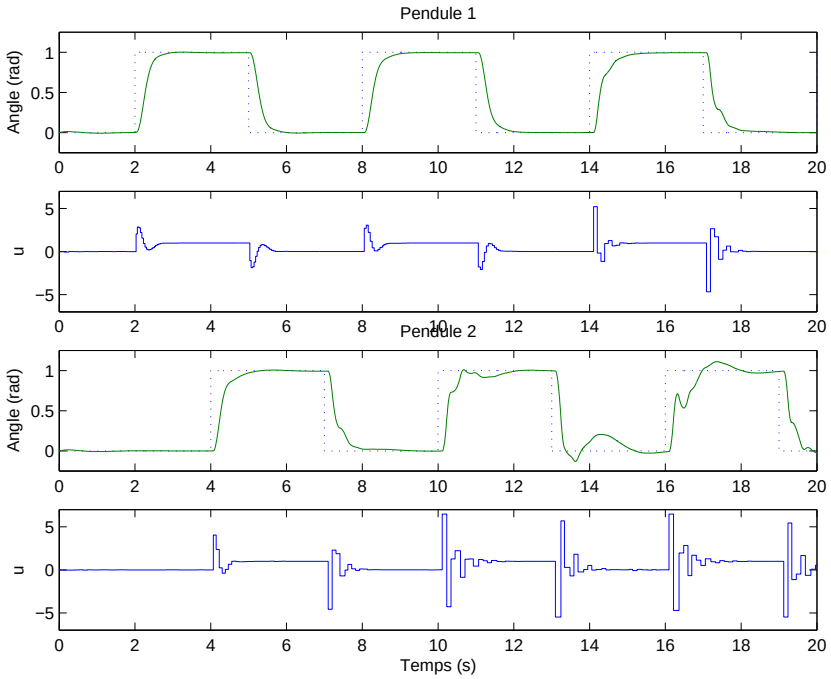


FIG. 6.4 – Evolution des pendules (objectifs constants)

On observe que le temps de réponse du premier pendule varie très faiblement quelle que soit la période. Une très légère perturbation apparaît entre 17 et 18s. Par comparaison avec la figure 6.3, l'amplitude de sa commande u est plus élevée quand la période est grande. Nous avions déjà observé ce comportement au chapitre 4. La réponse du second pendule est quant à elle bien plus perturbée. Alors que le temps de réponse pourrait être similaire au premier

pendule, le régime transitoire est fortement perturbé. Ceci est du à la variation de la latence entrée/sortie de la seconde commande, induite par le plus grand nombre de préemptions subies par *pend2*. Notons enfin que l'amplitude de la commande u est plus élevée avec les objectifs constants.

Par comparaison avec la figure 6.3, où les perturbations temporelles sont les mêmes, le correcteur à objectifs constants semble moins robuste aux perturbations. Il est donc préférable d'adapter les objectifs de commande à la variation de la période. De plus, dans le but d'établir un lien entre la période et la performance de la commande, il nous semble plus intéressant de prévoir et de contrôler l'évolution de la performance en fonction de la période plutôt que de subir une dégradation de la robustesse plus ou moins facile à quantifier.

6.2 Ordonnancement adapté à la commande

La régulation de l'ordonnancement du paragraphe précédent apporte une flexibilité vis à vis des ressources disponibles mais ne s'adapte pas aux variations de la qualité de la commande. Dans ce paragraphe nous proposons l'ébauche d'une structure d'ordonnancement qui permet une adaptation aux variations des paramètres suivants :

- le temps d'exécution,
- la consigne de charge processeur,
- la qualité de commande.

Contrairement au précédent régulateur d'ordonnancement, la répartition de charge entre les tâches n'est plus figée à la conception. Elle est maintenant dépendante de la qualité de commande pour privilégier le correcteur dont l'erreur est la plus élevée.

L'approche repose sur une extension du modèle de tâche élastique présenté par Buttazzo et al. (1998) où l'objectif est de répartir l'usage du processeur entre n tâches, en agissant sur leur période, sous les contraintes suivantes :

- la charge totale est inférieure à une référence U_d : $\sum_{i=1}^{n} U_i \leq U_d$,
- la période d'une tâche est bornée : $h_{i_{min}} \leq h_i \leq h_{i_{max}}$,
- la répartition de la charge est pondérée par des poids k_i.

L'ordonnancement est adapté en ligne en réponse à un changement de la référence de charge U_d, à une variation du temps d'exécution d'une tâche ou à la variation d'un poids k_i. La mise à jour de l'ordonnancement est obtenue par un algorithme itératif décrit dans Buttazzo et al. (1998) qui répartit l'usage U_d du processeur au prorata des poids k_i en tenant compte de la saturation des périodes.

Exemple L'exemple de la figure 6.5 comporte trois tâches dont le poids initial est $k_i = 1,\ i = 1 \ldots 3$. L'axe horizontal représente la répartition de la charge U_d entre les 3 tâches. Nous présentons 3 évolutions au cours du temps (axe vertical). Tout d'abord la réduction de

la consigne U_d réduit la charge disponible pour chaque tâche. Pour cela leur période respective est augmentée. Dans un deuxième temps le poids k_3 de la troisième tâche augmente, il en résulte que 2/4 de la charge U_d lui est assignée. Enfin dans un troisième temps, le poids k_3 est encore augmenté. Cette fois-ci la période de la tâche 1 étant à son maximum, il n'est plus possible d'en réduire la charge. Dans ce cas la tâche 3 dispose de 3/4 de la différence $U_d - U_1$ où U_1 est la charge minimale nécessaire à la tâche 1.

Ce comportement est assimilable à celui d'une chaîne de ressorts de longueur U_d où chaque ressort a un coefficient de raideur k_i et une longueur U_i bornée par $U_{i_{min}}$ et $U_{i_{max}}$. Dans notre cas les bornes ne sont pas sur la charge U_i mais sur la période. Il y a donc une subtilité mais rappelons que $U_i = c_i/h_i$ où c_i est le temps d'exécution et h_i la période. Pour plus de détails le lecteur peut se référer à l'article de Buttazzo et al. (1998).

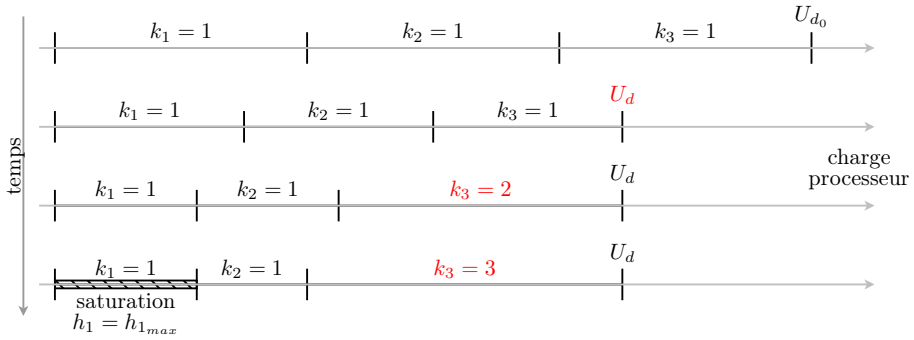


FIG. 6.5 – Exemple d'évolution de l'ordonnancement élastique

6.2.1 Approche proposée

Notre extension consiste à modifier en ligne le coefficient k_i d'une tâche en fonction d'une mesure de la qualité de sa commande. La structure est illustrée sur la figure 6.6. La difficulté est alors de choisir une mesure, accessible en ligne, qui exprime la qualité de la commande et de la lier au poids k_i.

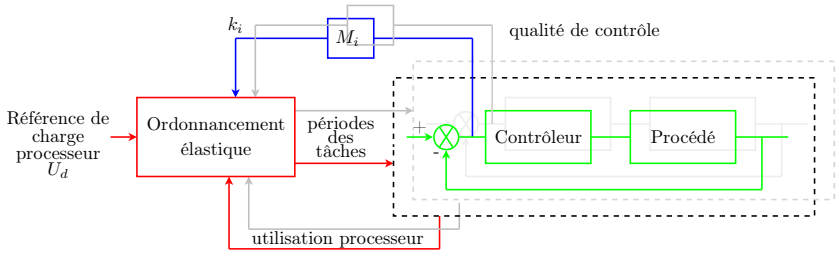


FIG. 6.6 – Illustration de la structure proposée

Pour ne pas surcharger le système, la période de la mise à jour de l'ordonnancement est supérieure à celle des commandes. La mesure doit donc refléter la qualité de la commande sur une période de l'ordonnancement. Nous proposons d'utiliser l'erreur quadratique moyenne de poursuite sur une fenêtre de temps d'une longueur égale à la période de l'ordonnancement. Dans ce cas, les tâches de commande sont modifiées car elles seules ont accès aux données nécessaires pour calculer cette moyenne.

Le coefficient k_i est défini en fonction de la mesure de qualité au travers du bloc M_i de la figure 6.6. Dans le cas le plus simple, M_i est un gain statique. Le choix des différents gain M_i doit produire des poids k_i similaires pour des commandes de qualité identique. Il faut cependant normaliser la mesure de qualité entre les procédés sinon un déséquilibre apparaît et favorise le procédé dont l'erreur absolue est la plus forte.

L'adaptation des poids k_i en fonction de la qualité de la commande constitue une contre réaction. Il est alors logique de s'interroger sur sa stabilité et sa dynamique. Pour cela il est nécessaire d'établir la relation qui relie les poids k_i à la qualité des commandes. Cette relation est particulièrement complexe. En effet elle contient le comportement de l'ordonnancement élastique et celui des commandes des procédés. L'ordonnancement est un système non linéaire statique où les périodes des tâches sont fonctions de la référence de charge, des poids k_i, des saturations sur les périodes et des temps d'exécution. La relation entre la période d'une commande et l'erreur quadratique utilisée pour la mesure de la qualité est difficile à exprimer et dépend du procédé, de son contrôleur et des signaux exogènes. Notons que pour une période fixe, l'erreur quadratique varie en fonction des signaux exogènes et des perturbations du procédé.

Au regard de ces remarques, un choix rigoureux des gains est difficile et n'a pas encore été formulé. Nous allons cependant illustrer le principe d'adaptation de l'ordonnancement sur un exemple où les gains sont réglés empiriquement.

6.2.2 Exemple

L'objectif de l'exemple est de contrôler deux pendules, le premier est un pendule de la forme d'un "T" présenté au chapitre 5, le second est un pendule stable présenté au chapitre 4. Les contrôleurs sont implémentés sur un système temps réel préemptif à priorités fixes similaires à l'exemple précédent (voir 6.1). La qualité des commandes est mesurée via l'erreur quadratique moyenne de poursuite de la consigne d'angle, sur une période de l'ordonnancement.

Le scénario de simulation est le suivant. Les deux pendules ont une référence excitée par un signal sinusoïdal. A $t = 12s$ une tâche perturbatrice, de priorité intermédiaire entre celle des deux pendules, apparaît. Un bruit est appliqué sur la mesure du second pendule à $t = 20s$.

L'évolution de l'ordonnancement et celle des pendules sont présentées respectivement sur

les figures 6.7 et 6.8. Cette dernière contient également le tracé de l'erreur quadratique des deux pendules et un tracé représentant la répartition de la charge entre les deux tâches. Lorsque le tracé vaut 1, la totalité de la charge processeur U_d est assignée au contrôle du pendule T et inversement pour la valeur 0.

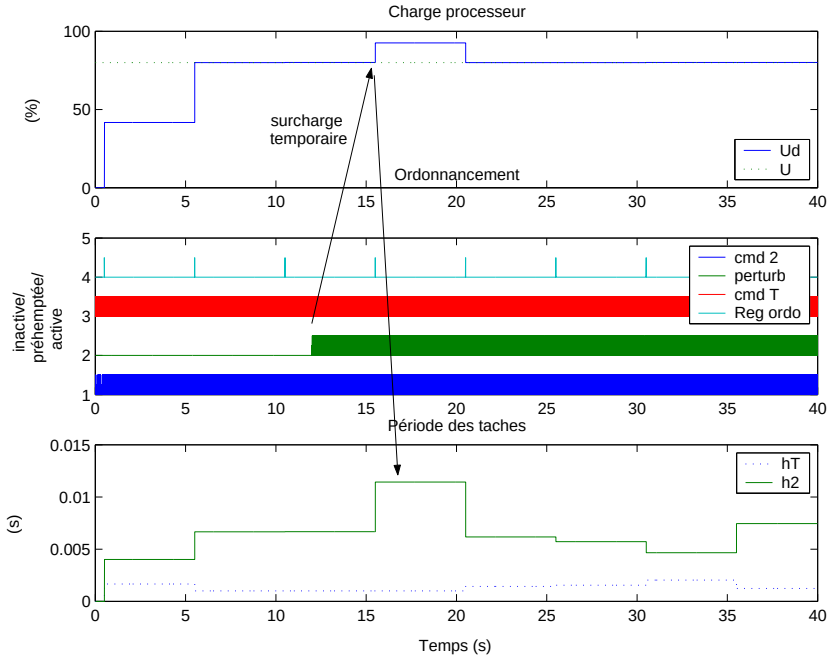


FIG. 6.7 – Evolution de l'ordonnancement

On observe tout d'abord que la surcharge due à l'apparition de la tâche perturbatrice est rejetée très rapidement, c'est à dire en une période de l'ordonnancement. Cette réjection est plus rapide qu'avec la régulation du paragraphe 6.1 car il n'y a pas de dynamique dans la structure proposée. Ce rejet a produit une forte hausse de la période de *cmd2* et une très faible pour *cmdT*. Ceci s'explique par la répartition de charge qui, à $t = 16s$, est très favorable au pendule T. Le coefficient k_i de *cmdT* est alors beaucoup plus grand que celui de *cmd2*.

L'apparition du bruit à $t = 20s$ augmente l'erreur quadratique du pendule 2, ce qui rééquilibre la répartition de la charge processeur en diminuant la période de *cmd2* et en augmentant celle de *cmdT*.

Enfin l'augmentation de la période de *cmdT* implique une baisse de qualité entre $t = 20s$ et $t = 36s$. A $t = 36s$ l'erreur quadratique du pendule T augmente fortement ce qui modifie la répartition de charge à son avantage.

Cet exemple illustre la faisabilité de l'approche et l'importance du paramètre k_i. Lors des 12 premières secondes, ni l'ordonnancement ni les pendules ne sont perturbés, le système est alors dans un fonctionnement normal. Cependant la répartition de charge avantage très

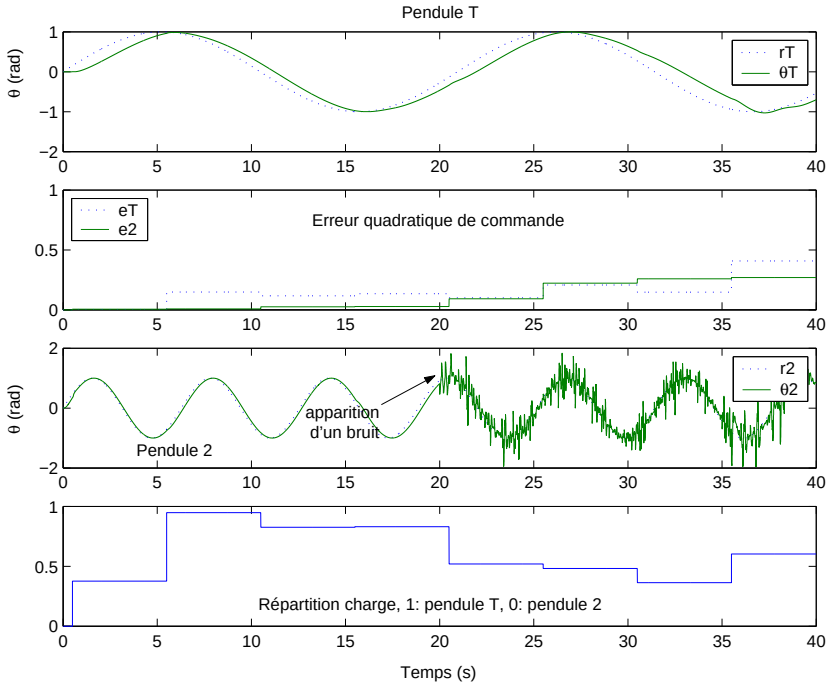


FIG. 6.8 – Evolution des pendules

fortement le pendule T. Ce déséquilibre ne semble pas logique pour un fonctionnement normal. Cela illustre la difficulté à normaliser la mesure de la qualité de commande entre différents procédés.

On peut constater, même si ce n'est pas le cas ici, des oscillations sur le tracé de la répartition de la charge. Dans des cas extrêmes on obtient des transitions brutales entre 0 et 1. Il semble important de contenir ces oscillations qui dépendent de plusieurs paramètres :

- La dynamique des procédés et la relation entre la période et la mesure de la qualité des commandes,
- la saturation des périodes,
- le gain d'adaptation des coefficients k_i, et le filtre s'il existe.

Il devient donc important d'établir un lien entre la période et la mesure de la qualité ce qui est une perspective de recherche.

Conclusion du chapitre

Ce chapitre a permis tout d'abord de regrouper dans un exemple d'application la régulation de l'ordonnancement et deux commandes à période variable. Le flexibilité apportée par l'ordonnancement régulé est appréciable tout en ayant des commandes dont le comportement est prévisible. Nous avons constaté une nouvelle fois que l'adaptation des objectifs

de commande à la période conduit à une meilleure robustesse.

Dans une seconde partie une nouvelle structure d'ordonnancement a été introduite où la mesure de la qualité des commandes permet de répartir les ressources processeur pour privilégier la commande dont la qualité doit être améliorée. L'exemple montre qu'il est nécessaire d'approfondir la connaissance de l'influence de la période sur la qualité de la commande.

Chapitre 7

Conclusion

Nous avons abordé dans ce travail quelques aspects de l'interaction entre l'ordonnancement et la commande, dans le but d'apporter plus de flexibilité vis à vis de la variation des ressources de calcul. Pour cela nous avons élaboré d'une part un ordonnancement qui s'adapte aux fluctuations des ressources de calcul. D'autre part, ce dernier devant modifier la période des tâches, nous avons développé une méthodologie de synthèse pour obtenir un correcteur à période variable.

Le premier chapitre est une introduction au contrôle d'un procédé physique par ordinateur. Après le rappel de la conception traditionnelle d'un tel système, les interactions entre l'ordonnancement et la commande sont énumérées et les solutions disponibles dans la littérature sont citées.

Le deuxième chapitre propose une modélisation et utilise un contrôleur linéaire pour réguler la charge processeur induite par un ensemble de tâches périodiques. Cela permet de s'adapter à la fluctuation des ressources disponibles, induite par les variations du temps d'exécution des tâches, de la puissance de calcul du processeur ou de la référence de charge.

Le chapitre trois est consacré à la présentation de la commande robuste H_∞ et des systèmes linéaires à paramètres variants (LPV). On y présente les critères pour quantifier la performance, la formulation du problème de synthèse d'un correcteur et enfin les outils de synthèse dédiés aux systèmes LPV à temps discret.

Dans le chapitre quatre, le commande robuste des systèmes LPV est utilisée pour concevoir un correcteur dont la période peut être modifiée en ligne. Pour cela nous présentons plusieurs modélisations de la discrétisation paramétrée d'un modèle linéaire continu afin d'obtenir un modèle LPV à temps discret. A l'issue de quoi l'application des outils de synthèse précédemment introduits produit un correcteur à temps discret dont le paramètre variant est sa période.

Le chapitre cinq illustre, par la commande d'un procédé physique instable et assez difficile à contrôler, la faisabilité expérimentale de la commande à période variable que nous

proposons.

Enfin le dernier chapitre regroupe dans une même application, la régulation de l'ordonnancement et des commandes à période variable. Le bénéfice de l'approche est la flexibilité apportée vis à vis de la fluctuation de l'usage des ressources tout en gardant prévisible le comportement des commandes. Dans la même situation, un ordonnancement à paramètres prédéfinis conduirait soit à une sous-utilisation du processeur s'il est bien dimensionné, soit à de fortes dégradations de la commande s'il est choisi au plus juste.

A l'issue de ce travail, les perspectives à envisager sont les suivantes :

La régulation de l'ordonnancement du deuxième chapitre doit être complétée. Elle est focalisée sur la charge sans tenir compte d'autres paramètres comme la latence entrée/sortie ou le dépassement d'échéances qui ont pourtant des effets importants sur les tâches de commande. Bien qu'une baisse de la charge diminue qualitativement les perturbations temporelles, la simple régulation de la charge n'est pas suffisante. Dans l'objectif d'un usage optimal des ressources de calcul tout en garantissant des performances de commande, l'ordonnancement doit tenir compte d'une mesure en ligne de la qualité de la commande. L'idée proposée dans le dernier chapitre peut être un bon point de départ. Le plus important mais aussi le plus difficile est d'exprimer la mesure de qualité en fonction de l'ordonnancement et de la commande du procédé.

D'autre part la synthèse de correcteurs dynamiques à période variable doit être enrichie. Ce type de commande peut être utile là où la période fluctue et notamment en réponse à la variation des capacités de calcul d'un processeur mais aussi à celles d'un canal de communication. Il serait utile de réduire le conservatisme de l'approche polytopique, dû à l'approximation d'une courbe pour un polytope mais aussi à l'usage d'une matrice de Lyapunov constante. Enfin le formalisme basé sur la transformation linéaire fractionnelle, qui a de nombreux atouts en commande robuste, permettrait de prendre en compte des incertitudes en plus de la variation de la période. Il est donc important de poursuivre l'effort dans cette direction et pour cela il faut d'abord résoudre les problèmes numériques rencontrés lors de nos essais.

Annexe A

Publications

Cette annexe contient les 3 papiers suivants :

- Synthesis of a Sampling Period Dependent Controller using LPV Approach, D. Robert, O. Sename et D. Simon, *Proceedings of the 5th IFAC Symposium on Robust Control Design*, Toulouse, France, July, 2006
- Sampling Period Dependent RST Controller used in Control/Scheduling co-Design, D. Robert, O. Sename et D. Simon, *Proc of the 16th IFAC World Congress*, Praha, Czech Republic , July, 2005
- Robust control/scheduling co-design : application to robot control, D. Simon, D. Robert et O. Sename, *Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium*, San Francisco, United States, March, 2005

SAMPLING PERIOD DEPENDENT RST CONTROLLER USED IN CONTROL/SCHEDULING CO-DESIGN

David Robert * **Olivier Sename** * **Daniel Simon** **

* *Laboratoire d'automatique de Grenoble, ENSIEG, BP 46 38402 Saint Martin d'Heres Cedex, France*

** *INRIA Rhone-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint-Ismier Cedex, France*

Abstract: The paper presents a sampling period dependent RST controller used in the context of control/scheduling co-design. This plant controller is implemented on a real-time operating system as a control task. A scheduling controller adapts control task periods to regulate the processor load. The link established between scheduling and plant control provides more flexibility and robustness w.r.t. the variation of the processor load. Parameterized RST and scheduling controller synthesis are presented and a co-design example illustrates the approach flexibility.
Copyright© 2005 IFAC.

Keywords: Computer controlled systems, digital control, RST controller, feedback scheduling

1. INTRODUCTION

Digital control systems are often implemented as a set of tasks running on top off a real-time operating system. Control tasks are generally viewed by the scheduling community as hard real-time tasks with fixed sampling periods and known "worst case execution time" (WCET). On the other hand control community supposes that periods are constants and can not be changed on-line. Theses assumptions are often too restrictive. For instance many control laws can tolerate small variations on the sampling period without leading to instability. On the other hand WCET based scheduling of control laws with varying execution time leads to an under utilization of the processor unit. Moreover, practical estimation of the WCET is a difficult and time consuming work. Finally embedded applications with complex control laws require flexibility to allocate computational resources on the fly.

In this paper a feedback scheduler regulates the processor utilization to avoid overload. It acts on the task periods and measures task loads. Plant control laws, computed by the tasks, should fit the sampling period variations, therefore a sampling period dependant RST controller is proposed. The link established between scheduling and plant control provides more flexibility and robustness w.r.t. the variation of the processor load.

Control and scheduling co-design is a recent interest in both the computer and control communities. A first approach uses off-line co-design. In (Seto *et al.*, 1996) optimal periods which maximize the control law performances w.r.t. the resource constraints are obtained by solving an optimization problem. In (Ryu and Hong, 1998)

control law performances depend on the periods as well as on input/output latencies and an heuristic algorithm computes the optimal periods. In (Palopoli *et al.*, 2002) an optimization problem is solved to find optimal task periods and feedback gains. The previous approaches do not bring flexibility therefore on-line adaptation was studied. In (Cervin and Eker, 2000) a feedback controller with a sampling period dependant PID controller is used. In (Cervin *et al.*, 2002; Eker *et al.*, 2000) rescaling factors obtained by off-line optimization preserve the optimality of a set of control task periods. In (Caccamo *et al.*, 2000; Sha *et al.*, 2000) a modified Constant Bandwidth Server adapts task periods to locally handle an overrun when the execution time varies. In (Cervin and Eker, 2003) a framework, based on a Constant Bandwidth Server, is used to enhance the determinism of the co-design. In (Sename *et al.*, 2003) the authors have developed a LQ controller to regulate the processor load and a delay dependant robust controller to stabilize the plant w.r.t. latencies. In the computer community, Wei and Yu (2003) and Lu *et al.* (2002) use two PID controllers to regulate the deadline miss-ratio and the CPU consumption of real-time tasks.

The outline of this paper is as follow. Section 2 describes the scheduling model and its controller design. Section 3 presents a parameterized polynomial pole placement synthesis used for the plant control. In the section 4 constant and variable plant controller performances are compared. An illustrative example of the co-design is presented in section 5. Finally, the paper ends with some conclusions and further research directions.

2. SCHEDULING CONTROLLER

Plant control task periods are on-line adjusted according to the processor load variations. This work is done by a specific task, the scheduling controller, which has the highest priority and a period much larger than the others tasks.

Figure (1) presents the bloc diagram of the feedback scheduling. The scheduling is viewed as a dynamical system which output is the processor utilization and inputs are control task frequencies. As far as the adaptation of the control tasks is concerned, the load of the other tasks is seen as an output disturbance.

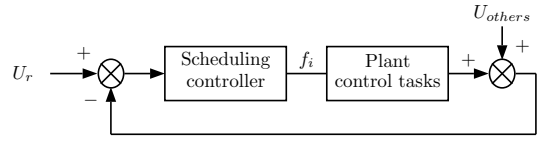


Fig. 1. Feedback scheduling bloc diagram

The scheduling is here limited to periodic tasks. In this case the processor load induced by a task is defined by $U = \frac{c}{h}$ where c and h are the execution time and period of the task. It can be rewritten as $U = c\ f$, where f is the task frequency, which is linear when the execution time is constant.

Based on this equation, processor load induced by a task is modelled in a way similar to Cervin *et al.* (2002). Thus, for each period h_s of the scheduling controller, the processor load of one plant control task is estimated as :

$$\hat{U}_{kh_s} = \lambda\, \hat{U}_{(k-1)h_s} + (1-\lambda)\, \overline{c}_{kh_s}\ f_{(k-1)h_s} \quad (1)$$

where f is the sampling frequency currently assigned to the plant control task and $\overline{c}$ is the mean of its measured job execution-time. λ is a forgetting factor used to smooth the measure.

As $\overline{c}$ depends on the runtime environment (e.g. processor speed) a "normalized" linear model of the task i (2) is used for the scheduling controller synthesis where $\overline{c}$ is omitted and will be compensated by on-line gain-scheduling $(1/\overline{c})$, as illustrated by the figure (4) in section 5.

$$G_i(z) = \frac{\hat{U}_i(z)}{f_i(z)} = \frac{1-\lambda_i}{z-\lambda_i} \quad (2)$$

Finally, the scheduling of n plant control tasks is modelled by the discrete-time state-space representation :

$$\begin{cases} x(k+1) &= Ax(k) + Bf(k) \\ \hat{u}(k) &= Cx(k) \end{cases} \quad (3)$$

with $A = diag\{\lambda_1, \ldots, \lambda_n\}$, $B = diag\{1-\lambda_1, \ldots, 1-\lambda_n\}$ and $C = [1 \ldots 1]$. f and x are respectively the vectors of task frequencies and task loads whereas $\hat{u}$ is the load of all plant control tasks.

Based on this state-space representation standard control methodology can be used to design a controller. Here a discrete H_∞ synthesis is proposed and illustrated in section 5.

3. PLANT CONTROL DESIGN

As the scheduling controller adapts the periods of the plant control tasks, the latter should fit the new sampling period h in order to preserve stability. The design objective is to obtain an unique controller as a function of h instead of a map of different controllers. Thus the stability can be theoretically ensured for all h. A polynomial pole-placement approach is used as explained below.

The structure of the plant controller, in figure (1), is a well-known two degrees of freedom discrete-

time controller. The desired closed-loop performances are specified by model matching, as done in (Åström and Wittenmark, 1997).

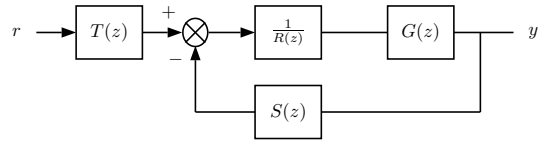


Fig. 2. Plant controller bloc diagram

As h is not constant, it is easier to specify the closed-loop model in a continuous-time form. Adding a dependance on h allows it to be parameterized by the sampling period (as emphasized in section 4). The plant and closed-loop models are expressed by the transfer functions :

$$G(s) = \frac{B(s)}{A(s)} \text{ and } G_m(s,h) = \frac{B_m(s,h)}{A_m(s,h)} \quad (4)$$

A formal discretization, exact or approximative, with h as parameter, leads to both discrete-time transfer functions $G(z,h)$ and $G_m(z,h)$ as below, with $i \leq j$:

$$G(z,h) = \frac{B(z,h)}{A(z,h)} = \frac{b_i(h)z^i + ... + b_0(h)}{z^j + a_{j-1}(h)z^{j-1} + ... + a_0(h)} \quad (5)$$

For easier reading the dependence in the z and h variables will sometimes be missed out. Then, the closed-loop controlled system of figure (2) can be expressed by :

$$G_{cl}(z,h) = \frac{B(z,h)\,T(z,h)}{A(z,h)\,R(z,h) + B(z,h)\,S(z,h)} \quad (6)$$

The synthesis objective is to find $R(z,h)$, $S(z,h)$ and $T(z,h)$ such as $G_{cl}(z,h)$ matches $G_m(z,h)$. In a way similar to Åström and Wittenmark (1997) the following factorizations (7) are used, where A and B do not have any common factors, A^+ and B^+ are monic and contained stable poles and zeros which can be cancelled, A^- and B^- contained unstable poles and zeros, R_d and S_d specify given factors of R and S (e.g. integral terms).

$$\begin{aligned} A &= A^+A^- & S &= A^+S_dS' \\ B &= B^+B^- & R &= B^+R_dR' \\ B_m &= B^-B_m' & T &= A^+T' \end{aligned} \quad (7)$$

The matching problem (8) needs to solve the Diophantine equation (9). R, S and T are obtained with (7). A_o is the observer polynomial which will be defined, see (Åström and Wittenmark, 1997).

$$\frac{B^+A^+B^-T'}{B^+A^+(A^-R_dR' + B^-S_dS')} = \frac{B^-B_m'}{A_m} \quad (8)$$

$$(A^-R_dR' + B^-S_dS') = A_mA_o \quad (9)$$

$$T' = B_m'A_o \quad (10)$$

Necessary and sufficient conditions to obtain an unique solution and a causal controller, when computation time is disregarded, are given on polynomial degrees in z, by : for all $h \in \mathbb{R}^+$,

$$d^oA_m - d^oB_m = d^oA - d^oB \quad (11)$$

$$d^oR = d^oS = d^oT \quad (12)$$

$$d^oA_o = d^oA + d^oS_d + d^oR_d + d^oA^- - d^oB^+ - 1 - d^oA_m \quad (13)$$

The internal observer should be faster than the closed-loop, therefore the roots of A_o are chosen in accordance with A_m and (13). The observer dynamic can be defined as a function of the closed-loop dynamic (e.g. five times faster) or arbitrary defined by a continuous-time model discretized as for the closed-loop model.

Remark 1. Due to the dependence in h, equation (9) has to be solved analytically, which can be done with any symbolic computation software.

Remark 2. h-dependent plant model of equation (5) could also be the result of an interpolation of identifications issue at different sampling periods.

Remark 3. Conditions (11-13) should be generically satisfy, i.e. for almost all values of h. In practice the designer can select a set of h for which none of polynomials in (11-13) loses a degree in z.

4. CONSTANT VS VARIABLE PLANT CONTROLLER PERFORMANCES

As introduced in section 3 the plant controller can be designed with constant or sampling period dependent performance specifications. Both approaches are here compared in an example.

4.1 Plant controller synthesis

The plant is a stable pendulum. The continuous model (14), linearized at the equilibrium, expressed the angular position in function of the applied torque with $\omega_0 = 3.77\ rad/s$, $\xi = 0.2$ and $K = \omega_0/9.81$, see (Eker *et al.*, 2000).

$$G(s) = \frac{K}{s^2 + 2\xi\omega_0 s + \omega_0^2} \quad (14)$$

Closed-loop performances are defined in continuous-time form by (15) with $\xi_m = 0.7$ and $K_m = \omega_m^2$ for a unity static gain. Internal observer A_o is chosen five times faster than the closed-loop model therefore $\omega_{obs} = 5\ \omega_m$ in (16). A zero steady state error to a step disturbance is required.

$$G_m(s,h) = \frac{K_m(h)}{s^2 + 2\xi_m\omega_m(h)s + \omega_m^2(h)} \quad (15)$$

$$A_o(s,h) = s^2 + 2\xi_m\omega_{obs}(h)s + \omega_{obs}^2(h) \quad (16)$$

Both cases are considered :

- Constant closed-loop performances
  $\omega_m = 10\ rad/s$
- Variable closed-loop performances
  According to the rule of thumb of Åström and Wittenmark (1997), $\omega_{obs}\ h \approx 0.2 \ldots 0.6$ because ω_{obs} is the highest pulsation of the closed-loop system. By choosing the middle of the interval, $\omega_m = 1/5\ \omega_{obs} = 4/(50\ h)$.

Then $G(s)$ and $G_m(s,h)$ are discretized with Tustin's approximation. Two discrete-time transfer functions are obtained as :

$$G(z,h) = \frac{K_d(h)\ (z+1)^2}{z^2 + a_{1d}(h)\ z + a_{0d}(h)} \quad (17)$$

By defining $B^- = (z+1)^2$, which appears in both G and G_m numerators, $B^+ = 1$, $A^+ = 1$, $S_d = 1$ and $R_d = (z-1)$ for the integral action, equation (9) and (10) are solved. Using conditions (12) and (13) we obtain $degT = degS = degR = 2$, $degR' = 1$ and $degA_0 = 2$. Solving the Diophantine equation (9) leads to the three polynomials (18) where each parameters are expressed by a fourth order rational function in h.

$$R(z,h) = (z-1)\ (r_1(h)\ z + r_0(h)) \quad (18)$$

$$S(z,h) = s_2(h)\ z^2 + s_1(h)\ z + s_0(h) \quad (19)$$

$$T(z,h) = t_2(h)\ z^2 + t_1(h)\ z + t_0(h) \quad (20)$$

4.2 Simulation

The two plant control methods designed in the previous section are now compared for the sampling periods 2, 8, 15, 25, and 50 ms using a non-linear model of the pendulum.

Figure (3,a) presents the step response when closed-loop performance specifications are constant. The system becomes unstable for sampling periods greater than 25 ms. According to the rule of thumb $\omega_{obs}\ h \approx 0.2 \ldots 0.6$, a good sampling period for this case is $h \in [4; 12]\ ms$. It appears that this rule is here more restrictive than necessary which is due to a minimum phase margin guaranteed by the rule.

Figure (3,b) presents the step response when closed-loop performances depend on the sampling period. As it was predictable, the system is never unstable and the performances (i.e. ω_m) decrease when the sampling period increases.

To conclude this section, adapting closed-loop performance specifications according to the sampling period can increase robust stability w.r.t. sampling period changes. The next section presents the benefits of adapting closed-loop performances in the context of control/scheduling co-design.

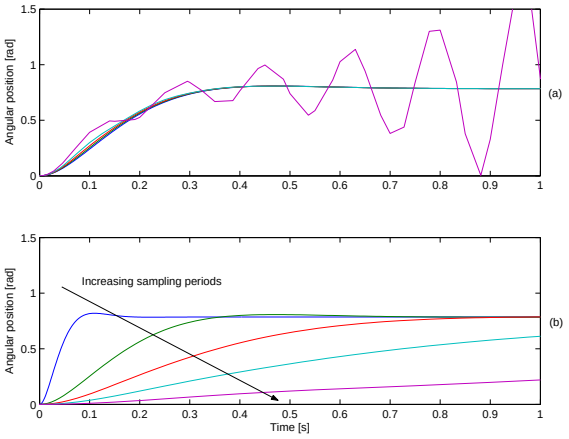


Fig. 3. Step response of the closed-loop with constant (a) or variable (b) performance specifications for the sampling periods 2 to 50ms

5. CO-DESIGN EXAMPLE

In this example, two independent stable pendulums are controlled by a computer. Four tasks share the same processor unit on top of a real-time operating system with priority based scheduling. Two tasks control the both pendulums, one task implements the scheduling controller and the last is a disruptive task. Tables (1) and (2) summarize scheduling and pendulum properties respectively. Pendulum control laws are designed as explained in the previous section.

Table 1. Scheduling properties

Task	Priority	Period (ms)	Execution time(ms)
schedctrl	1	500	1
pend1	2	4 to 400	2
disturb	3	4	2
pend2	4	4 to 400	2

Table 2. Plant properties

Pendulum	Task	ξ	ω_o	ξ_m	ω_m
1	pend1	0.2	3.77	0.7	10
2	pend2	0.2	4.08	0.7	10

5.1 Scheduler control design

The scheduling controller has to adjust the two plant control task periods when the desired processor utilization varies, in response to a reference change or a disruptive task.

Figure (4) presents the bloc diagram of this example which includes saturations and on-line gain-scheduling used to compensate the variations of the job execution time, see section 2.

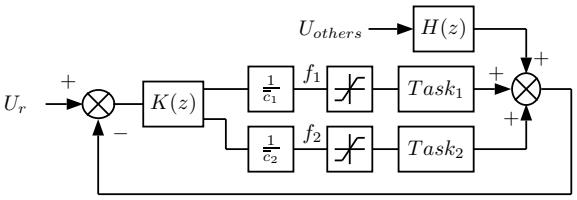


Fig. 4. Control scheme for CPU resources

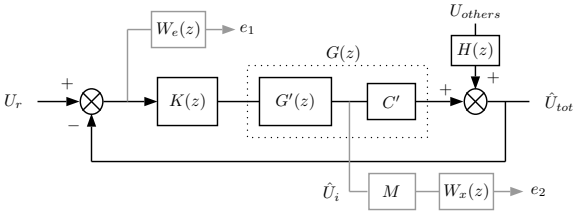


Fig. 5. H_∞ design bloc diagram

The bloc diagram of figure (5) is considered for the H_∞ design where $G'(z)$ is defined by (3) with $C = I_2$ to obtain a full state output and $C' = [1\ 1]$. $H(z)$ is the dynamic of the sensor which measures the load of the other tasks. It can be a first order filter as in (1) but not necessarily. The template W_e specifies the performances on the load tracking error as follows :

$$W_e(s) = \frac{s/M_s + \omega_b}{s + \omega_s \epsilon} \tag{21}$$

with $M_s = 2$, $\omega_s = 1\ rad/s$, $\epsilon = 0.05$ to obtain a closed-loop rise time of 3 s, a static error less than 5 % and a good robustness margin. Matrix $M = [-\alpha\ 1]$ and template W_x allow to specify the load allocation between the two control tasks. With a large gain in W_x, $U_2 - \alpha\, U_1 \approx 0$ i.e. $\frac{U_2}{U_1} \approx \alpha$.

All templates are discretized with a sampling period of 500 ms. Finally discrete-time H_∞ synthesis, using the LMI Control Toolbox of Matlab, produces a discrete-time scheduling controller of order 3.

5.2 Co-simulation

This co-simulation is done using a non-linear model of the pendulum and the Truetime Matlab Toolbox, see (Cervin *et al.*, 2003). A specific Simulink bloc simulates a real-time kernel with tasks, scheduler and input/output capabilities used to connect it with classical Simulink dynamic system blocs.

Two examples are presented, the first with two pendulums with constant closed-loop performances and the second with variable ones.

The scheduling scenario is the same for both examples. Processor utilization starts with a reference of 80 %. At time $t = 5\ s$ reference decreases to 60 %, representing a decrease of available resources. At time $t = 12\ s$ a new (disruptive) task appears which needs 50 % of the processor resources. The priorities and the load allocation ratio $\alpha = 2$ are chosen to favor the first pendulum. The two pendulums are excited by square wave reference signal of period 6 s.

5.3 Example 1, constant closed-loop performances

In this example, the two control laws have constant closed-loop performance specifications summarized in the table 2. A good sampling period for these performances is $h \in [4; 12]\ ms$, see section 4.

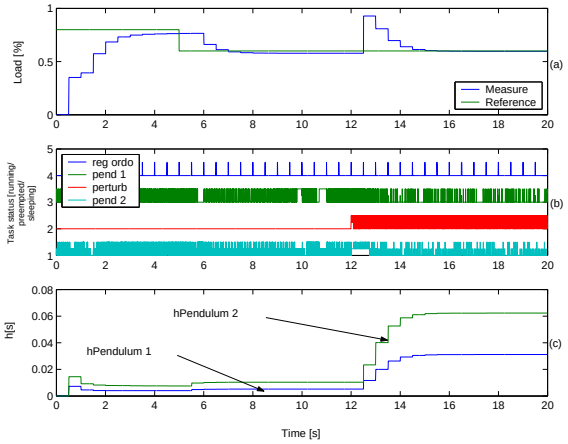


Fig. 6. Example 1, scheduling : (a) processor load, (b) scheduling timing, (c) control task periods

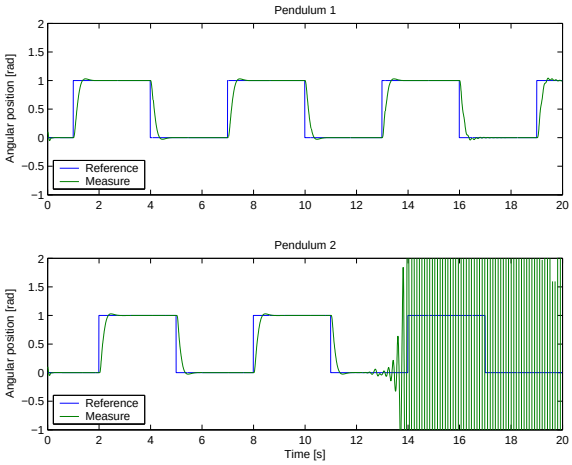


Fig. 7. Example 1, pendulum positions

In figure (6), the periods of the two pendulum control laws are adjusted when the processor load reference changes ($t = 5\ s$) or when load disturbance appears ($t = 12\ s$). Due to the load allocation ratio, the two periods are not identical. As the execution time is the same for the two control tasks (in this example) the allocation ratio can be seen on the periods, for example at $t = 18\ s$ $\frac{h_2}{h1} \approx 2 = \alpha$.

In figure (7), both pendulums remain stable after the load reference change. The pendulum two becomes unstable when the disruptive task appears because the sampling period becomes too high.

5.4 Example 2, varying closed-loop performances

In this example, pendulum 1 (high priority) has constant performances and pendulum 2 (low priority) has varying performances, see section 4. The scheduling behavior is the same as the first example, therefore it is not shown here.

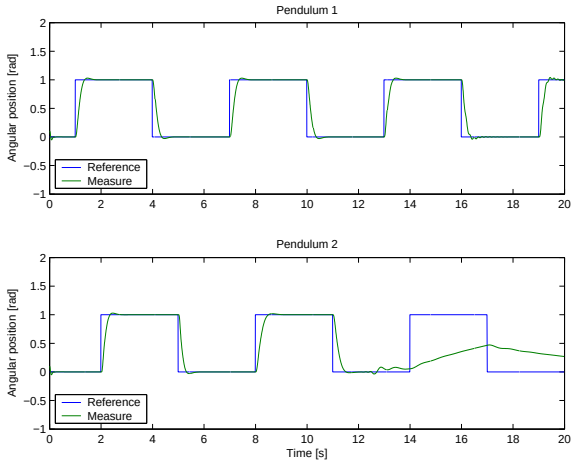


Fig. 8. Example 2, pendulum positions

In figure (8) both pendulums stay stable even if the sampling periods rise sharply at $t = 12\ s$. The stability of the pendulum 2 is preserved by decreasing the performances. This example emphasizes the interest for adapting closed-loop performances w.r.t. sampling period.

6. CONCLUSION AND FUTURE WORK

In this paper a processor load regulation has been presented based on a simple scheduling model and H_∞ synthesis. The synthesis of a sampling period dependant RST controller for the plant has been exposed. Then it has been shown that a co-design, with these two controllers and varying plant performances, can improve robustness in stability and flexibility w.r.t. processor load variations.

However this work is only at its first stage. The scheduling model use a measure of execution time which may be difficult to get in practice, especially with off-the-self operating systems. In this case an observer should be design to estimate the execution time. On the other hand the RST synthesis procedure is complex to used with higher order plant due to the high order, in h, of the R, S and T polynomial parameters. The next studies will use parameter dependent controller synthesis such as LPV.

REFERENCES

Åström, K.J. and B. Wittenmark (1997). *Computer-Controlled Systems*. Information and systems sciences series. 3rd ed.. Prentice Hall. New Jersey.

Caccamo, M., G. Buttazzo and L. Sha (2000). Elastic feedback control. In: *Proc. 12th Euromicro Conference on Real-Time Systems*. Stockholm, Sweden.

Cervin, A. and J. Eker (2000). Feedback scheduling of control tasks. In: *Proc. 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Cervin, A. and J. Eker (2003). The Control Server: A computational model for real-time control tasks. In: *Proc. 15th Euromicro Conference on Real-Time Systems*. Porto, Portugal.

Cervin, A., D. Henriksson, B. Lincoln, J. Eker and K.E. Årzén (2003). How does control timing affect performance?. *IEEE Control Systems Magazine* **23**(3), 16–30.

Cervin, A., J. Eker, B. Bernhardsson and K.E. Årzén (2002). Feedback-feedforward scheduling of control tasks. *Real-Time Systems* **23**(1–2), 25–53.

Eker, J., P. Hagander and K.E. Årzén (2000). A feedback scheduler for real-time controller tasks. *Control Engineering Practice* **8**(12), 1369–1378.

Lu, C., J.A. Stankovic, S.H. Son and G. Tao (2002). Feedback control real-time scheduling: Framework, modeling and algorithms. *Real-Time Systems Journal* **23**(1/2), 85–126.

Palopoli, L., C. Pinello, A. Sangiovanni-Vincentelli, L. Elghaoui and A. Bicchi (2002). Synthesis of robust control systems under resource constraints. In: *Hybrid Systems: Computation and Control*. Standford.

Ryu, M. and S. Hong (1998). Toward automatic synthesis of schedulable real-time controllers. *International Journal of Integrated Computer-Aided Engineering* **5**(3), 261–277.

Sename, O., D. Simon and D. Robert (2003). Feedback scheduling for real-time control of systems with communication delays. In: *Proc. IEEE Confer. on Emerging Technologies and Factory Automation*. Lisbon, Portugal.

Seto, D., J.P. Lehoczky, L. Sha and K.G. Shin (1996). On task schedulability in real-time control systems. In: *Proc. 17th IEEE Real-Time Systems Symposium*. Washington, DC.

Sha, L., X. Liu, M. Caccamo and G. Buttazzo (2000). Online control optimization using load driven scheduling. In: *Proc. 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Wei, L. and H. Yu (2003). Research on a soft real-time scheduling algorithm based on hybrid adaptive control architecture. In: *Proc. 2003 American Control Conference*.

SYNTHESIS OF A SAMPLING PERIOD DEPENDENT CONTROLLER USING LPV APPROACH

David Robert * **Olivier Sename** * **Daniel Simon** **

* *Laboratoire d'automatique de Grenoble, ENSIEG, BP 46 38402 Saint Martin d'Heres Cedex, France*

** *INRIA Rhone-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint-Ismier Cedex, France*

Abstract: This study comes within the context of the adaptation to processor/network load variations. We develop first a parameterized discrete-time plant representation w.r.t. the sampling period. An H_∞ approach is then considered such that the bandwidth of the weighting functions depends on the sampling period. The linear parameter-varying (LPV) design of Apkarian *et al.* (1995) is then used to get a sampling period dependent discrete-time controller. An academical example illustrates the proposed approach. *Copyright© 2005 IFAC*

Keywords: Robust control, linear parameter varying controller, digital control

1. INTRODUCTION

Control laws are today implemented using digital processors. Some applications like cars, household appliances are subject to economical constraints which lead to low-cost computing unit. Low-cost implies constrained computation and network resources. In this context designers try to use all the resources optimally. A solution is to improve the flexibility of the system by on-line adaptation of the processor/network utilization. Adjusting processor or network load of a control law can be obtained by changing the algorithm or adapting the sampling period. This paper deals with the latter case and presents the synthesis of a control law with varying sampling period.

Adaptation of control law to resource variations is a recent research interest. In (Cervin and Eker, 2000) a feedback controller with a sampling period dependant PID controller is used. In (Cervin *et al.*, 2002; Eker *et al.*, 2000) rescaling factors obtained by off-line optimization preserve the optimality of a set of control task periods. In (Sename *et al.*, 2003) a LQ controller regulates the processor load and a bank of delay-dependant robust controller stabilizes the plant w.r.t. latencies. In (Robert *et al.*, 2005) design of a sampling period dependent RST controller was proposed.

The presented contribution uses linear parameter-varying (LPV) approach of the robust linear control framework. The main point is the problem formulation such that it can be solved following the LPV design of Apkarian *et al.* (1995). We propose a parameterized discretization of the continuous time plant and the weighting functions, leading to a discrete-time sampling period dependent augmented plant. Then LPV design produces a discrete-time sampling period dependent controller.

The outline of this paper is as follows. Section 2 recalls the LPV design of Apkarian *et al.* (1995).

Section 3 describes the plant discretization. Section 4 deals with the specification of closed-loop objectives. Section 5 comments briefly the augmented plant. An illustrative example is presented in section 6. Finally, the paper ends with some conclusions and further research directions.

2. BACKGROUND

The results of Apkarian *et al.* (1995) are here recalled.

Let the discrete-time LPV plant, mapping exogenous inputs w and control inputs u to controlled outputs z and measured outputs y, with $x \in \mathbb{R}^n$, be given by the polytopic model

$$\begin{cases} x_{k+1} = A(\theta)x_k + B_1(\theta)w + B_2(\theta)u \\ z = C_1(\theta)x_k + D_{11}(\theta)w + D_{12}(\theta)u \\ y = C_2(\theta)x_k + D_{21}(\theta)w + D_{22}(\theta)u \end{cases} \quad (1)$$

where the dependence of $A(\theta)$, $B(\theta)$, $C(\theta)$ and $D(\theta)$ on θ is affine and the parameter vector θ, ranges over a fixed polytope Θ with r vertices w_i

$$\Theta = \left\{ \sum_{i=1}^{r} \alpha_i \omega_i : \alpha_i \geq 0, \sum_{i=1}^{r} \alpha_i = 1 \right\} \quad (2)$$

Proposition 1. Under the assumptions :

(A1) $D_{22}(\theta) = 0$
(A2) $B_2(\theta), C_2(\theta), D_{12}(\theta), D_{21}(\theta)$ are parameter-independent
(A3) the pairs $(A(\theta), B_2)$ and $(A(\theta), C_2)$ are quadratically stabilizable and detectable over θ respectively,

the self-scheduled controller

$$\begin{cases} x_{K_{k+1}} &= A_K(\theta)x_{K_k} + B_K(\theta)y_k \\ u_k &= C_K(\theta)x_{K_k} + D_K(\theta)y_k \end{cases} \quad (3)$$

where $x_K \in \mathbb{R}^n$, ensures over all parameter trajectories, for the closed-loop system of figure (1):

- closed-loop quadratic stability
- $\mathscr{L}_2$-induced norm of the operator mapping w into z bounded by γ, i.e. $\|z\|_2 < \gamma\|w\|_2$

if and only if there exist γ and two symmetric matrices (R, S) satisfying $2r + 1$ LMIs

$$\left(\begin{array}{c|c} N_R & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i R A_i^T - R & A_i R C_{1i}^T & B_{1i} \\ C_{1i} R A_i^T & -\gamma I + C_{1i} R C_{1i}^T & D_{11i} \\ \hline B_{1i}^T & D_{11i}^T & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_R & 0 \\ \hline 0 & I \end{array}\right) < 0, i = 1 \ldots r \quad (4)$$

$$\left(\begin{array}{c|c} N_S & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i^T S A_i - S & A_i^T S B_{1i} & C_{1i}^T \\ B_{1i}^T S A_i & -\gamma I + B_{1i}^T S B_{1i} & D_{11i}^T \\ \hline C_{1i} & D_{11i} & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_S & 0 \\ \hline 0 & I \end{array}\right) < 0, i = 1 \ldots r \quad (5)$$

$$\begin{pmatrix} R & I \\ I & S \end{pmatrix} \geq 0 \quad (6)$$

where A_i, B_{1i}, C_{1i}, D_{11i} are $A(\theta)$, $B_1(\theta)$, $C_1(\theta)$, $D_{11}(\theta)$ evaluated at the i^{th} vertex of the parameter polytope. N_S and N_R denote bases of null spaces of (B_2^T, D_{12}^T) and (C_2, D_{21}) respectively.

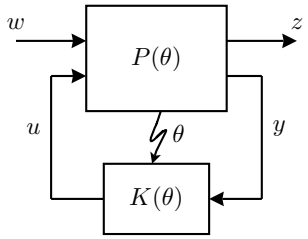


Fig. 1. Closed-loop of the LPV system

Once R, S and γ are obtained, the controllers are reconstructed at each vertex of the parameter polytope. The self-scheduled controller $K(\theta)$ is then the convex combination of these controllers

$$K(\theta) : \begin{pmatrix} A_K(\theta) & B_K(\theta) \\ C_K(\theta) & D_K(\theta) \end{pmatrix} = \sum_{i=1}^{r} \alpha_i \begin{pmatrix} A_{K_i} & B_{K_i} \\ C_{K_i} & D_{K_i} \end{pmatrix} \quad (7)$$

$$\text{with } \alpha_i \text{ such that } \theta = \sum_{i=1}^{r} \alpha_i \omega_i \quad (8)$$

Remark 1. This synthesis uses a constant Lyapunov function approach which is known to produce a sub-optimal controller.

3. PARAMETERIZED DISCRETIZATION OF THE PLANT

We consider state space representation of continuous time plants as :

$$G : \begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \quad (9)$$

Exact discretization of this system with a zero order hold at the sampling period h can be computed using expression (10) and (11), see (Åström and Wittenmark, 1997).

$$\begin{pmatrix} A_d & B_d \\ 0 & I \end{pmatrix} = exp(\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} h) \quad (10)$$

$$C_d = C \quad D_d = D \quad (11)$$

Our objective is to obtain the discrete-time LPV system (12)

$$G_d : \begin{cases} x_{k+1} &= A_d(h)\, x_k + B_d(h)\, u_k \\ y_k &= C_d(h)\, x_k + D_d(h)\, u_k \end{cases} \quad (12)$$

with h ranging in $[h_{min}; h_{max}]$. However the dependence on h of this model should match the requirements of section 2. In (10) A_d and B_d are not affine on h. We propose to approximate the exponential by a Taylor series of order N as :

$$e^{Mh} \approx \sum_{i=0}^{N} \frac{(Mh)^i}{i!}, \quad (13)$$

which leads, with $H = [h\ h^2\ \ldots\ h^N]$, to

$$A_d(h) \approx I + \sum_{i=1}^{N} \frac{A^i}{i!} h^i := A_d(H) \quad (14)$$

$$B_d(h) \approx \sum_{i=1}^{N} \frac{A^{i-1}B}{i!} h^i := B_d(H) \quad (15)$$

Now dependence on H is affine. To get a polytope containing H, a solution is to choose Θ with the 2^N vertices w_i corresponding to the vertices of the hypercube (17).

$$\Theta = \left\{ \sum_{i=1}^{2^N} \alpha_i \omega_i : \alpha_i \geq 0, \sum_{i=1}^{2^N} \alpha_i = 1 \right\} \quad (16)$$

$$\{h, h^2, \ldots, h^N\}, \; h \in [h_{min}; h_{max}] \quad (17)$$

This leads to the plant polytopic model (18) where G_{d_i} are $G_d(H)$ evaluated at the vertices w_i.

$$G_d(\theta) = \sum_{i=1}^{2^N} \alpha_i G_{d_i} \quad \text{and} \quad \theta = \sum_{i=1}^{2^N} \alpha_i w_i \quad (18)$$

As the self-scheduled controller will be a convex combination of 2^N "vertex" controllers (7), the choice of the series order N is a compromise between the approximation accuracy and the controller complexity. The criteria (19) is used to evaluate the approximation error. The H_∞ norm is chosen here to express the worst case error between G_{d_e} and G_d, two discretizations which use matrix exponential and Taylor series approximation of order N respectively.

$$J_N = \max_{h_{min}<h<h_{max}} \| G_{d_e}(h,z) - G_d(h,z) \|_\infty \quad (19)$$

Remark 2. Note that exact calculation of matrix exponential via diagonalisation or Cayley-Hamilton theorems are here more involved as their expression will lead to non affine representations of $A_d(h)$ and $B_d(h)$.

Remark 3. In (14), h, h^2, ..., h^N are viewed as independent parameters which leads to some conservatism. To lower it, the number of polytope vertices could be reduced but we have postponed it as a future work.

Remark 4. Taylor approximation, especially of low order, is inaccurate for large value of the parameter h. Therefore it is important to split (13) as $e^{Mh} = e^{M(h_0+\delta_h)} = e^{Mh_0}e^{M\delta_h}$ with $\delta_h \in [h_0 - h_{min}; h_{max} - h_0]$, then calculate exactly e^{Mh_0} and apply approximation only on $e^{M\delta_h}$ (see section 6).

4. OBJECTIVE SPECIFICATION

In the H_∞ framework, the general control configuration of figure 2 is considered, where W_i and W_o are weighting functions specifying closed-loop performances (see (Skogestad and Postlethwaite, 1996)). The objective is then to find a controller K such internal stability is achieved and $\|\tilde{z}\|_2 < \gamma \|\tilde{w}\|_2$.

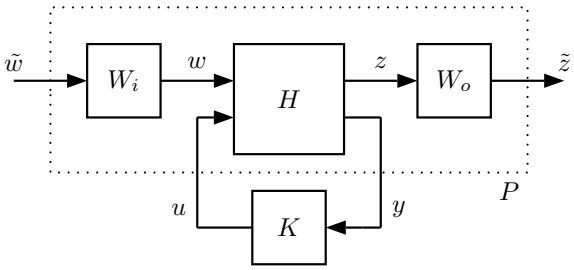


Fig. 2. Focused interconnection

Classical control design assumes constant performance objectives and produces a controller with an unique sampling period. This sampling period is chosen according to the controller bandwidth, the noise sensibility and the availability of computation resources. If the sampling period varies then the usable controller bandwidth varies and closed-loop objectives should logically be adapted. Therefore we propose to adapt the bandwidth of the weighting functions. In this aim, W_i and W_o are split into two parts :

- the constant part with constant poles and zeros. This allows, for instance, to compensate for oscillations or flexible modes which are, by definition, independent of the sampling period. This part is merged with the plant before its discetization.
- the variable part contains the highest poles and zeros whose pulsations are expressed as an affine function of the frequency $f = 1/h$. This permits to adapt the bandwidth of the weighting functions. These poles and zeros are here constrained to be *real* by the discretization step.

To discretize the variable part $V(s)$ of a weighting function, we propose the following methodology

(1) factorize $V(s)$ as a product of first order systems, with $a_i, b_i \in \mathbb{R}$

$$V(s) = \alpha \prod_i \frac{s - b_i f}{s - a_i f} = \alpha \prod_i V_i(s) \quad (20)$$

(2) use the observable canonical form for $V_i(s)$

$$V_i(s) : \begin{cases} \dot{x}_i = a_i f \; x_i + f(a_i - b_i) \; u_i \\ y_i = x_i + u_i \end{cases} \quad (21)$$

(3) form the series interconnection of the state space representation of each $V_i(s)$ and thus

$$V(s) : \begin{cases} \dot{x}_v = A_v f \; x_v + B_v f \; u_v \\ y_v = C_v \; x_v + D_v \; u_v \end{cases} \quad (22)$$

(4) discretize the state space representation of $V(s)$. Thanks to the affine dependence in f in (22) the discrete-time model of the variable part becomes independent of h as :

$$\begin{cases} A_{v_d} = e^{A_v f\, h} = e^{A_v} \\ B_{v_d} = (A_v f)^{-1}(A_{v_d} - I)B_v f \\ C_{v_d} = C_v \text{ and } D_{v_d} = D_v \end{cases} \quad (23)$$

Remark 5. The serial interconnection of two systems $V_i(s)$ leads to the expressions (24). It is easy to verify that even with more than two first order systems, matrices A and B of $V(s)$ remain affine in f :

$$\begin{aligned} A_v &= \begin{pmatrix} a_1 & 0 \\ a_2 - b_2 & a_2 \end{pmatrix} & B_v &= \begin{pmatrix} a_1 - b_1 \\ a_2 - b_2 \end{pmatrix} \\ C_v &= \begin{pmatrix} 1 & 1 \end{pmatrix} & D_v &= 1 \\ x_v &= (x_1 \; x_2)^T \end{aligned} \quad (24)$$

Remark 6. The simplification between f and h in (23) makes easy the discretization step. This is why plant and weighting functions are separately discretized and the augmented plant is obtained in discrete time afterwards.

5. AUGMENTED PLANT AND SYNTHESIS

Interconnection of figure 2 between the discrete-time polytopic model of the plant and the weighting functions leads to the discrete-time LPV augmented plant $P(\theta)$,

$$P(\theta) = \left(\begin{array}{ccc|cc} A(\theta) & B_w(\theta)C_i & 0 & B_w(\theta) & B_u(\theta) \\ 0 & A_i & 0 & B_i & 0 \\ B_oC_z & B_oD_{zw}C_i & A_o & B_oD_{zw}D_i & B_oD_{zu} \\ \hline D_oC_z & D_oD_{zw}C_i & C_o & D_oD_{zw}D_i & D_oD_{zu} \\ C_y & D_{yw}C_i & 0 & D_{yw}D_i & D_{yu} \end{array}\right) \quad \text{with } \theta \in \Theta \quad (25)$$

In (25) assumption (A2) is not satisfied due to $B_u(\theta)$ term in $B_2(\theta)$. To avoid this, a strictly proper filter is added on the control input. It is a numerical artefact, therefore its bandwidth should be chosen high enough to be negligible regarding the plant and objective bandwidths.

Then the controller is designed as explained in section 2. On-line scheduling of the controller needs the computation of α_i knowing h which is easy with the chosen polytope Θ.

6. ILLUSTRATIVE EXAMPLE

In this section the proposed methodology is illustrated on a simple example. The plant is a non-linear stable pendulum. For the control design, linearization at the equilibrium leads to a continuous time second order system with $\omega_0 = 3.13$ rad/s and $\xi = 0.47$.

$$\begin{cases} \dot{x}_G = \begin{pmatrix} 0 & 1 \\ -\omega_0 & -2\xi\omega_0 \end{pmatrix} x_G + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u \\ y_G = \begin{pmatrix} 1 & 0 \end{pmatrix} x_G \end{cases} \quad (26)$$

The sampling period interval is $\mathbb{H} = [4; 40]$ ms which allows to divide by 10 the processor/network load induced by the control law.

Figure 3 presents the absolute error between the exact and the approximated plant discretization. Here, the approximation location h_0 is the middle of $\mathbb{H}$. One can easily see its impact on the error. To obtain an error less than 10^{-5} we chose $N = 3$ as the Taylor series order.

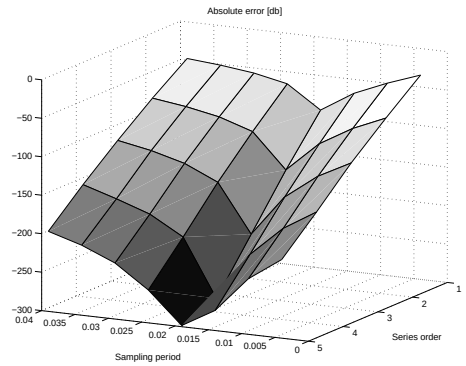


Fig. 3. Absolute error between exact and approximated discretization

6.1 Controller design

Closed-loop objectives are specified as a classical H_∞ problem of figure 4 where, corresponding to figure 2, W_o is the diagonal concatenation of W_S, W_{KS}, W_T and $W_i = I$. Following section

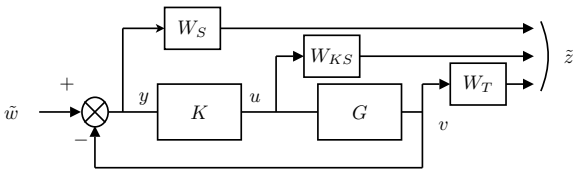


Fig. 4. H_∞ problem block diagram

4, usual weighting functions (see (Skogestad and Postlethwaite, 1996)) are expressed with a varying bandwidth by

$$W_S = \frac{s/M_S + \omega_S}{s + \omega_S\epsilon_S} = \frac{y}{\tilde{w}} \quad (27)$$

$$W_{KS} = \frac{s + \omega_{KS}/M_{KS}}{\epsilon_{KS}s + \omega_{KS}} = \frac{u}{\tilde{w}} \quad (28)$$

$$W_T = \frac{\epsilon_T s + \omega_T}{s + \omega_T/M_T} = \frac{v}{\tilde{w}} \quad (29)$$

with $\omega_s = 0.1\ \omega_e, M_S = 5, \epsilon_S = 0.001$, $\omega_{KS} = 0.3\ \omega_e, M_{KS} = 20, \epsilon_{KS} = 0.2$ and $\omega_T = 0.2\ \omega_e, M_T = 1, \epsilon_T = 0.1$, $\omega_e = 2\pi\ f$.

Interconnection of the polytopic plant model and the weighting functions leads to the following augmented plant with 8 vertices :

$$P(\theta) = \left(\begin{array}{cc|cc} A(\theta) & 0 & B_w(\theta) & B_u(\theta) \\ B_o(\theta)C_z & A_o(\theta) & B_o(\theta)D_{zw} & B_oD_{zu} \\ \hline D_oC_z & C_o & D_oD_{zw} & D_oD_{zu} \\ C_y & 0 & D_{yw} & 0 \end{array}\right)$$

Applying a low-pass filter on the control input, all assumptions of proposition 1 are satisfied. Using the previous LPV design, the problem is solved and we obtain $\gamma = 46$.

Now the closed-loop behavior is studied for ten sampling periods chosen in $\mathbb{H}$. Figures 5 presents the sensitivity functions where solid lines are the discrete-time specified weighting functions and dot lines the obtained ones. First, care must be taken with the discretization of the continuous-time weighting functions. Here the maximum of $1/W_S(z)$ have been cut down due to a zero at high frequencies in $W_S(s)$ but with no consequences. Then, in the three plots, sensitivity functions have variable bandwidth which satisfies quite well the specified ratio of 10. However there is an offset between specified and measured bandwidth due to the high value of γ. The last plot is the step response of the continuous-time non-linear model controlled by this discrete-time control law. In accordance with the sensibility functions, the speed varies with the sampling period and the variation of the settling time satisfies quite well the specified ratio of 10.

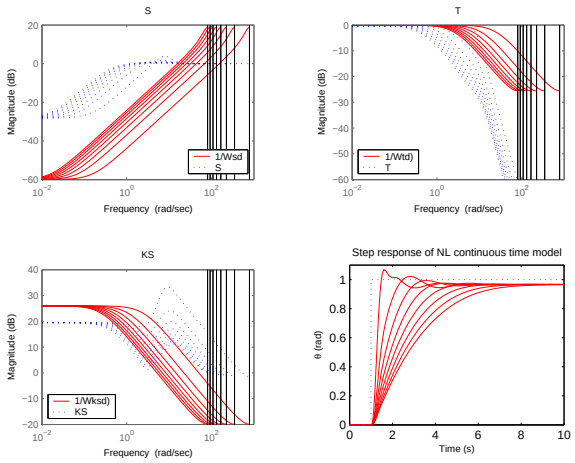


Fig. 5. Sensibility functions and step response of the LPV design

Now the LPV controller is compared to ten LTI controllers, each one synthesized for a specific sampling periods. LTI controllers are obtained by a discrete-time synthesis using the LPV's weighting functions in accordance with the period and applied on an exact discretization of the plant model. The optimal γ decreases from 28 to 8, when the period increases from $4ms$ to $40ms$, whereas it stayed at 46 for the LPV synthesis. This difference illustrates the conservatism of the LPV synthesis. On figure 6, the LTI sensitivity functions have a higher crossover pulsation but with a variation less than 10. Logically the rise time of the step response is higher but the overshoot penalizes the settling time.

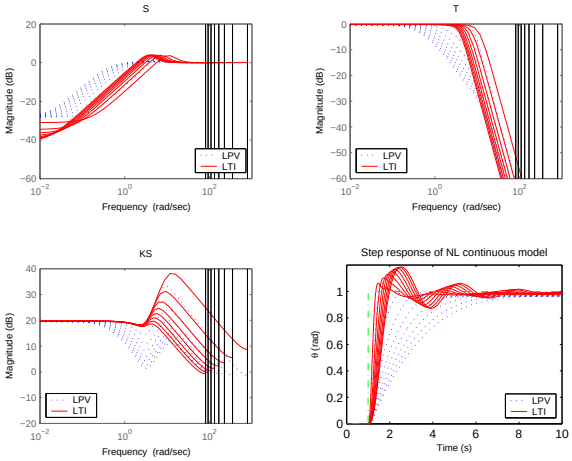


Fig. 6. Comparaison between the LPV and LTI designs

6.2 On-line self-scheduling

Two controller implementations are compared : the LPV controller and a bank of ten LTI controllers, previously presented. Each controller of this bank is continuously computed and the applied control input is chosen according to the current sampling period. This is a "controller blending".

Differences between these two solutions are resources needed to compute/transmit the control input and the stability during the switching phase. The LPV controller computes only one control law and ensures stability for all parameter trajectories. Overload is needed when the parameter varies because of the new convex combination computation. The bank of LTI controllers implies a high processor/network load because all controllers are computed in parallel and many measures and control inputs are transmitted. Moreover, without extra care, stability is not guaranteed during a controller switch. Therefore, even if controller blending is a simple and well used solution in gain scheduling applications, it is a very bad solution in the resources sharing context.

Now step responses with a variation of the sampling period are presented. In figure 7 the sampling period h varies from 40 ms to $4ms$, at $t = 3$ s, during the transient period of the plant. The decrease of h should imply performance increase of the LPV controller, which is verified by the slope variation of the time response. The LTI controllers, although speeder, are disrupted by the switch and need a larger control input to remain stable, as expressed by the quadratic sum of the control input appended to the plot legend.

In figure 8 a smoother variation is applied. Steps

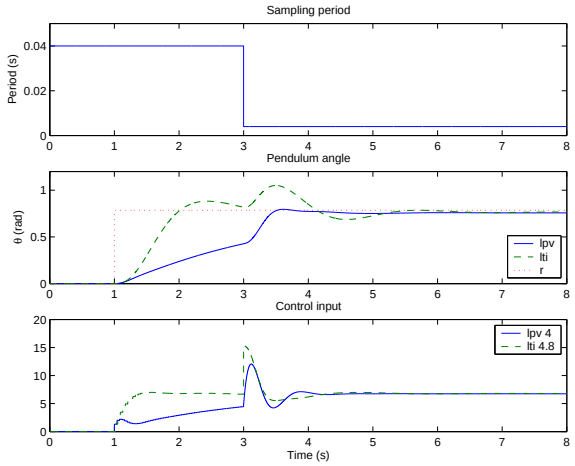


Fig. 7. Step response with step variation of the sampling period

on the sampling period plot express switching of the LTI controllers as there are in finite number. The LPV controller becomes slower as expressed

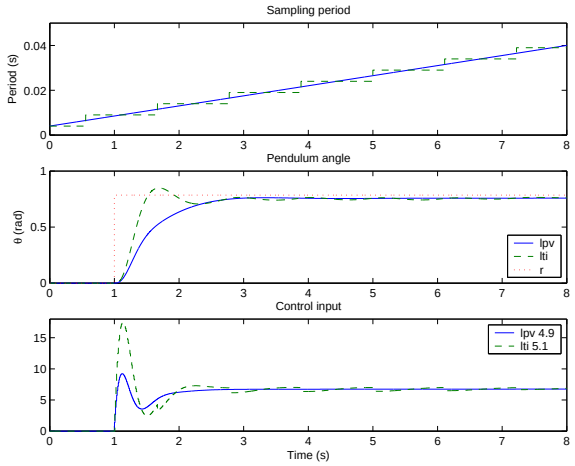


Fig. 8. Step response with ramp variation of the sampling period

by the slope variation of the time response. Small variations on the control input show that the blending controller is disrupted at each switch and thus uses much the control input.

Based on this simulations, the LPV controller produces a smoother control input and needs less energy. Although the conservatism of the LPV synthesis, the settling times are here similar. This emphasis the need to tune each LTI controller which is an extra work with regards to the LPV synthesis.

However, implementation of LPV controllers may be complex, especially when they are synthesized in continuous-time because of the on-line discretization needed when the parameter changes (see (Apkarian, 1997)). In our case, the "vertex controllers" are designed in discrete-time and only their convex combination have to be computed.

7. CONCLUSION

In this paper, an LPV approach is proposed to design a discrete-time linear controller with a varying sampling period and varying performances. This control law can be useful in the context of adaptation to varying processor or network load where a bank of switching controllers would need too much ressources.

The LPV scheme allows here to guarantee the closed-loop quadratic stability, a bounded $\mathscr{L}_2$-induced norm for all variation of the sampling period and have a predictable closed-loop behavior. To reduce the conservatism of the proposed approach, extensions using smaller polytopes, parameter-dependent Lyapunov function should be studied.

REFERENCES

Apkarian, P. (1997). On the discretization of LMI-synthesized linear parameter-varying controllers. *Automatica* **33**(4), 655–662.

Apkarian, P., P. Gahinet and G. Becker (1995). Self-scheduled H_∞ control of linear parameter-varying systems: A design example. *Automatica* **31**(9), 1251–1262.

Åström, K.J. and B. Wittenmark (1997). *Computer-Controlled Systems*. Information and systems sciences series. 3rd ed.. Prentice Hall. New Jersey.

Cervin, A. and J. Eker (2000). Feedback scheduling of control tasks. In: *Proceedings of the 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Cervin, A., J. Eker, B. Bernhardsson and K. Årzén (2002). Feedback-feedforward scheduling of control tasks. *Real-Time Systems* **23**(1–2), 25–53.

Eker, J., P. Hagander and K. Årzén (2000). A feedback scheduler for real-time controller tasks. *Control Engineering Practice* **8**(12), 1369–1378.

Robert, D., O. Sename and D. Simon (2005). Sampling period dependent RST controller used in control/scheduling co-design. In: *Proceedings of the 16th IFAC World Congress*. Prague, Czech Republic.

Sename, O., D. Simon and D. Robert (2003). Feedback scheduling for real-time control of systems with communication delays. In: *Proceedings of the 2003 IEEE ETFA*. Lisbon, Portugal. pp. 454–61.

Skogestad, S. and I. Postlethwaite (1996). *Multivariable Feedback Control: analysis and design*. John Wiley and Sons.

Robust control/scheduling co-design: application to robot control

Daniel Simon
INRIA Rhône-Alpes
655 avenue de l'Europe, Montbonnot
38334 Saint-Ismier Cedex, France
Daniel.Simon@inrialpes.fr

David Robert, Olivier Sename
Laboratoire d'Automatique de Grenoble
ENSIEG-BP 46
38402 Saint Martin d'Hères Cedex, France
{david.robert,olivier.sename}@inpg.fr

Abstract

Control systems running on a computer are subject to timing disturbances coming from implementation constraints. Fortunately closed-loop systems behave robustly w.r.t. modelling errors and disturbances, and the controller design can be performed to explicitly enhance robustness against specific uncertainties. On one hand robustness in process controllers can be used to comply with weakly modelled timing uncertainties. On the other hand the principle of robust closed-loop control can also be applied to the real-time scheduler to provide on-line adaption of some scheduling parameters, with the objective of controlling the computing resource allocation. The control performance specification may be set according to both control and implementation constraints. The approach is illustrated through several examples using simulation and an experimental feedback scheduler is briefly described.

Keywords: control/scheduling co-design, feed-back scheduling, robustness, resource management

1. Introduction

Digital control systems can be implemented as a set of tasks running on top of an off-the-shelf real-time operating system (RTOS) using fixed-priority and preemption. The performance of the control, e.g measured by the tracking error, and even more importantly its stability, strongly relies on the values of the sampling rates and sensor-to-actuator latencies (the latency we consider for control purpose is the delay between the instant when a measure q_n is taken on a sensor and the instant when the control signal $U(q_n)$ is received by the actuators [1]). Therefore it is essential that the implementation of the controller respect an adequate temporal behaviour to meet the expected performance. However implementation constraints such as multi-rate sampling, preemption, synchronisation and various sources of delays makes the run-time behaviour of the controller very difficult to accurately predict. However as we deal with closed-loop controllers we may take advantage of the *robustness* of such systems to design and implement flexible and adaptive real-time control architectures.

This paper deals with robust and adaptive solutions for real-time scheduling and control co-design. In the next section we review some properties of closed-loop controllers in contrast with real-time implementation constraints. Some recent results in control and scheduling co-design are recalled in section 3. Section 4 gives an overview of a new feedback scheduling strategy aimed to on-line adapt the tasks period according to the computing resource activity. This approach is then applied in the design of a robot controller in section 5, for which the importance of integrated control/scheduling co-design is emphasised. Finally an experimental feedback scheduler implementation is described in section 6 and future research directions conclude the paper.

2. Problem position: background on control and implementation constraints

Closed-loop digital control systems use a computer to periodically sample sensors, compute a control law and send control signals to the actuators of a continuous time physical process. The control algorithm can be either designed in continuous time and then discretized or directly synthesised in discrete time taking account of a model of the plant sampled by a zero-order holder. Control theory for linear systems sampled at fixed rates has been established a long time ago, e.g. [1].

Assigning an adequate value for the sampling rate is a decisive duty as this value has a direct impact on the control performance and stability. While an abso-

lute lower limit for the sampling rate is given by Shannon's theorem, in practise rules of thumb are used to give a useful range of control frequencies according to the process dynamics and to the desired closed-loop bandwidth (see for example section 5.3). A general rule is that decreasing the control period and latencies allows for improved control performance, e.g. measured by the tracking error or disturbances rejection.

2.1. Digital control of continuous systems

To implement a controller the basic idea consist in running the whole set of control equations in a unique periodic real-time task which clock gives the controller sampling rate. In fact all parts of the control algorithm do not have an equal weight and urgency w.r.t. the control performance. To minimise the latency a control law can be basically implemented as two real-time blocks, the urgent one sends the control signal directly computed from the sampled measures while updating the state estimation or parameters can be delayed or even more computed less frequently [1].

In fact, a complex system involves sub-systems with different dynamics which must be further coordinated [18]. Assigning different periods and priorities to different blocks according to their relative weight allows for a better control of critical latencies and for a more efficient use of the computing resource [15]. However in such cases finding adequate periods for each block is out of the scope of current control theory and must be done through case studies, simulation and experiments.

2.2. Control and timing uncertainty

While timing uncertainties have an impact on the control performance they are difficult to be accurately modelled or constrained to lie inside precisely known bounds. Thus it is worth examining the sensitivity of control systems w.r.t. timing fluctuations.

Control systems are often cited as examples of "hard real-time systems" where jitter and deadline violations are strictly forbidden. In fact experiments show that this assumption may be false for closed-loop control. Any practical feedback system is designed to obtain some stability margin and robustness w.r.t. the plant parameters uncertainty. This also provides robustness w.r.t. timing uncertainties: closed-loop systems are able to tolerate some amount of sampling period and computing delays deviations, jitter and occasional data loss with no loss of stability or integrity, e.g. [4]: their behaviour can still be considered as correct as long as the sample-induced disturbances stay inside the performance specification bounds.

Therefore the hard real-time assumption can softened to better cope with the reality of closed-loop control. For example they can be changed for "weakly hard" constraints: absolute deadlines are replaced by statistical ones, e.g. the allowable output jitter compliant with the desired control performance or the number of allowed deadlines miss over a specified time window [2]. Note that to be fully exploited weakly hard constraints should be associated with a decisional process: tasks missing their deadline can be for example delayed, aborted or skipped according to their impact on the control law behaviour.

Finding the values of such weakly hard constraints for a given control law is currently out of the scope of current control theory in the general case. However the intrinsic robustness of closed-loop controllers allows for complying with softened timing constraints specification and flexible scheduling design.

2.3. Control and scheduling

Usually, real-time systems are modelled by a set of recurrent tasks assigned to one or several processors and a worst case response times technique is used to analyse fixed-priority real-time systems. Well known scheduling policies, such as Rate Monotonic for fixed priorities and EDF for dynamic priorities, assign priorities according to timing parameters, respectively sampling periods and deadlines. They are said to be "optimal" as they maximise the number of tasks sets which can be scheduled with respect of deadlines, under some restrictive assumptions. Unfortunately they are not optimised for control purpose.

They hardly take into account precedence and synchronisation constraints which naturally appear in a control algorithm. The relative urgency or criticality of the control tasks can be unrelated with the timing parameters. Thus, the timing requirements of control systems w.r.t. the performance specification do not fit well with scheduling policies purely based on schedulability tests. It has been shown through experiments, e.g. [4], that a blind use of such traditional scheduling policy can lead to an inefficient controller implementation; on the other hand a scheduling policy based on application's requirements, associated with a right partition of the control algorithm into real-time modules may give better results.

Another example of unsuitability between computing and control requirements arises when using priority inheritance or priority ceiling protocols to bypass priority inversion due to mutual exclusion, e.g. to en-

sure the integrity of shared data. While they are designed to avoid dead-locks and minimise priority inversion lengths, such protocols jeopardise at run-time the initial schedule which was carefully designed to meet control requirements. As a consequence latencies along some control paths can be largely increased leading to a poor control performance or even instability.

Finally off-line schedulability analysis rely on a right estimation of the tasks worst case execution time. Even in embedded systems the processors use caches and pipelines to improve the average computing speed while decreasing the timing predictability. Another source of uncertainty may come from some pieces of the control algorithm. For example, the duration of a vision process highly depends on incoming data from a dynamic scene. Also some algorithms are iterative with a badly known convergence rate, so that the time before reaching a predefined threshold is unknown (and must be bounded by a timeout). In a dynamic environment some control activities can be suspended or resumed and control algorithms with different costs can be scheduled according to various control modes leading to large variations in the computing load.

Thus real-time control design based on worst case execution time, maximum expected delay and strict deadlines inevitably leads to a low average usage of the computing resource.

3. Related work

Control/scheduling co-design This mainly concerns the integration of control performance knowledge in the scheduling parameters assignment. Indeed, once a control algorithm has been designed, a first job consists in assigning timing parameters, i.e. periods of tasks and deadlines, so that the controller's implementation satisfies the control objective. This may be done off-line or on-line.

In off-line control/scheduling co-design setting adequate values for the timing parameters rapidly falls into case studies based on simulation and experiments. For instance in [11] off-line iterative optimisation is used to compute an adequate setting of periods, latencies and gains resulting in a requested control performance according to the available computing resource and implementation constraints. Also in [12] the temporal requirements of the control system are described using complex temporal attributes (e.g. nominal period and allowed variations, precedence constraints...): this model is then used by an off-line iterative heuristic procedure to assign the scheduling parameters (e.g. priorities and offsets) to meet the constraints.

Concerning co-design for on-line implementation, recent results deal with varying sampling rates in control loops in the framework of linear systems: for example [13] show that, while switching between two stable controllers, too frequent control period switches may lead to unstability. Unfortunately most real-life systems are non-linear and the extrapolation of timing assignment through linearising often gives rough estimations of allowable periods and latencies or even can be meaningless. In fact, as it will be shown in the robot control application, the plant knowledge is necessary to get an efficient control/scheduling co-design.

Feedback scheduling Besides traditional assignment of fixed scheduling parameters more flexible scheduling policies have been investigated. Let us cite e.g. [3] where the elasticity of the tasks' periods enables for controlling the quality of service of the system as a function of the current estimated load. While such an approach is still working in open loop w.r.t. a controlled plant, the on-line combination the control performance and implementation constraints lead to the feedback scheduling approach.

This new approach has been initiated both from the real-time computing side [10] and from the control side [5, 7, 6]. The idea consists in adding to the process controller an outer sampled feedback loop ("scheduling regulator") to control the scheduling parameters as a function of a QoC (Quality of Control) measure. It is expected that an on line adaption of the scheduling parameters of the controller may increase its overall efficiency w.r.t. timing uncertainties coming from the unknown controlled environment. Also we know from control theory that closing the loop may increase performance and robustness against disturbances when properly designed and tuned (otherwise it may lead to instability).

Figure 1 gives an general overview of a feed-back scheduler where an outer loop (the *scheduling controller*) adapts in real-time the scheduling parameters from measurements taken on the computer's activity, e.g. the computing load . Besides this controller working periodically (at a rate larger than the sampling periods of the plant control tasks), the system's structure may evolve along a discrete time scale upon occurrence of events, e.g. for new tasks admission or exception handling. These decisional processes may be handled by another real-time task, the *scheduling manager*, which is not further detailed in this paper. Notice that such a manager may give a reference to the controller resource utilisation.

The design problem can thus be stated as control performance optimisation under constraint of available computing resources. Major studies result from [8, 4]

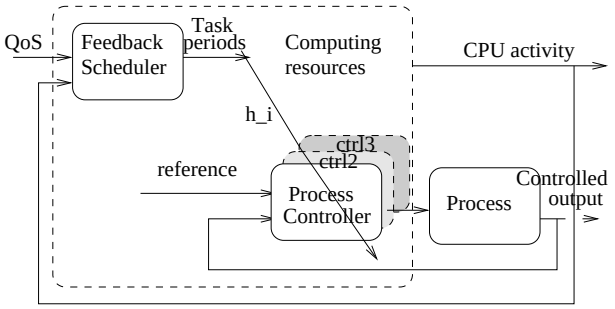


Figure 1. Hierarchical control structure

where it is suggested that a simple solution to this optimal control problem (i.e. under resource constraint) is the calculation of the new task periods by the rescaling:

$$h_i^{new} = h_{i_{nominal}} \frac{U}{U_{sp}}$$

where U_{sp} is the utilisation set-point. The feedback scheduler then controls the processor utilisation by assigning task periods that optimise the overall control performance.

Preliminary works have been done by the authors. In [14] an LQG approach is used to design the feedback scheduling while in [17], an H_∞ control problem is solved for a two tasks systems (without differentiation between both tasks). In what follows the proposed methodology is described.

4. A new methodology for Feed-back scheduling

Feedback scheduling is a dynamic approach allowing to better use the computing resources, in particular when the workload changes e.g. due to the activation of an admitted new task. Indeed, the CPU activity will be controlled according to the resource availability by adjusting scheduling parameters (i.e. period) of the plant control tasks.

In the approach here proposed, a way to take into account the resource sharing over a multitasks process is developed. In what follows, the control design issue is described including the control structure, the specification of control inputs and measured outputs, as well as the modelling step.

4.1. Control structure

In Fig 2 scheduling is viewed as a dynamical system between control task frequencies and processor utilisation. As far as the adaptation of the control tasks is concerned, the load of the other tasks is seen as an output disturbance.

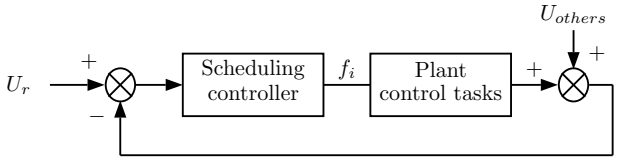


Figure 2. Feedback scheduling bloc diagram

4.2. Sensors and actuators

As stated in section 2.3, priorities must be assigned to control tasks according to their relative urgency ; this ordering remains the same in the case of a dynamic scheduler. Dynamic priorities, e.g. as used in EDF, only alter the interleaving of running tasks and will fail in adjusting the computing load w.r.t. the control requirements. In consequence we have elected the tasks periods to be the main actuators of the system running on top of a fixed priority scheduler[1].

As the aim is to adjust on-line the sampling periods of the controllers in order to meet the computing resource requirements, the control inputs are thus the periods of the control tasks.
The measured output is the CPU utilisation. Let us first recall that the scheduling is here limited to periodic tasks. In this case the processor load induced by a task is defined by $U = \frac{c}{h}$ where c and h are the execution time and period of the task. Hence processor load induced by a task is estimated, in a similar to way [6], for each period h_s of the scheduling controller, as :

$$\hat{U}_{kh_s} = \lambda\, \hat{U}_{(k-1)h_s} + (1-\lambda)\, \frac{\overline{c}_{kh_s}}{h_{(k-1)h_s}} \qquad (1)$$

where h is the sampling frequency currently assigned to the plant control task (i.e. at each sampling instant kh_s) and $\overline{c}$ is the mean of its measured job execution-time. λ is a forgetting factor used to smooth the measure.

4.3. Control design and implementation

The proposed control design method for feedback scheduling is here developed. First one should note

1 Possible secondary actuators are variants of the control algorithms, with different QoS contributions to the whole system. Such variants should be handled by the scheduling manager working on a discrete events time scale

that, as shown in [17], if the execution times are constant, then the relation, $U = \sum_{i=1}^{n} C_i f_i$ (where $f_i = 1/h_i$ is the frequency of the task) is a linear function (while it would not be as a function of the task periods). Therefore, using (1), the estimated CPU load is given as:

$$\hat{U}(kh_S) = \frac{(1-\lambda)}{z-\lambda} \sum_{i=1}^{n} \overline{c}_i(kh_S) f_i(kh_S) \qquad (2)$$

As $\overline{c}$ depends on the runtime environment (e.g. processor speed) a "normalised" linear model of the task i (i.e independent on the execution time) , G_i, is used for the scheduling controller synthesis where $\overline{c}$ is omitted and will be compensated by on-line gain-scheduling $(1/\overline{c})$ as shown below.

$$G_i(z) = \frac{\hat{U}(z)}{f_i(z)} = \frac{1-\lambda}{z-\lambda}, \quad \imath = 1, \ldots, n \qquad (3)$$

As illustration, in a single control task system, the control scheme is therefore as in figure 3 where the estimated execution-times are used on-line to adapt the gain of the controller for the original CPU system (2) (this allows to compensate the variations of the job execution time).

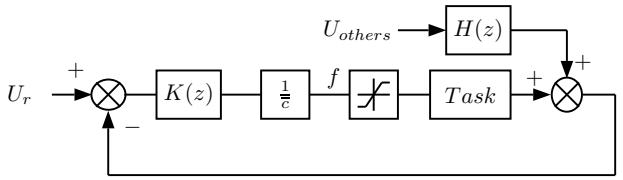


Figure 3. Control scheme for CPU resources

According to this control scheme, the design of the controller K can be made using any advanced control methodology. For the considered application (see section 5), we have chosen the well known H_∞ control theory which can lead to a robust controller w.r.t modelling errors (see [19] for details on H_∞ control). Moreover it provides good properties in the presence of external disturbance, as it is emphasised in the illustrative examples.

5. Integrated control-scheduling co-design in robot control

We consider here a seven degrees of freedom Mitsubishi PA10 robot arm that has been previously modelled and calibrated [16].

5.1. Plant modelling and control structure

The problem under consideration is to track a desired trajectory for the position of the end-effector. Using the Lagrange formalism the following model can be obtained:

$$\Gamma = M(q)\ddot{q} + Gra(q) + C(q, \dot{q}) \qquad (4)$$

where q stands for the positions of the joints, M is the inertia matrix, Gra is the gravity forces vector and C gathers Coriolis, centrifugal and friction forces.

The structure of the (ideal) linearising controller includes a compensation of the gravity, Coriolis/centrifugal effect and Inertia variations as well as a Proportional-Derivative (PD) controller for the tracking and stabilisation problem, of the form:

$$\Gamma = Gra(q) + C(q, \dot{q}) + K_p(q_d - q) + K_d(\dot{q_d} - \dot{q}), \quad (5)$$

leading to the linear closed-loop system $M(q)\ddot{q} = K_p(q_d - q) + K_d(\dot{q_d} - \dot{q})$.

This controller is divided in four tasks, i.e. a specific task is considered for the PD control, for the gravity, Inertia and Coriolis compensations, in order to use a multi-rate controller. In this first cautious feedback scheduling scheme, only the periods of the compensation tasks will be adapted, as they have a moderate impact on the closed-loop stability while they are more time consuming compared with the PD task.

5.2. A first step to co-design: evaluation of controller cost functions

In the co-design, the aim is to give the precedence (i.e. more resources) to the tasks that are more important for the robot control. To evaluate this importance a cost function is defined as

$$J = \int_0^{t_{trajectory}} \sum_{i=1}^{i=7} (q_i - qd_i)^2 dt, \qquad (6)$$

with q_i and qd_i the actual and desired positions of i^{th} joint. This cost function of figure 4 is here calculated off line first, for each subtask (among three) of the controller, according to the variation of the task periods (which may vary here from $0.5ms$ to $30ms$), the others remaining constant and fixed to $1ms$. Notice that we have chosen to let the PD control task at a constant period, due to its high influence in the robot control stabilisation strategy. As done in [14], a variable period could have been assigned to this task to emphasise its importance in the closed-loop stability. However, in practical robot applications, it will remain at a constant rate.

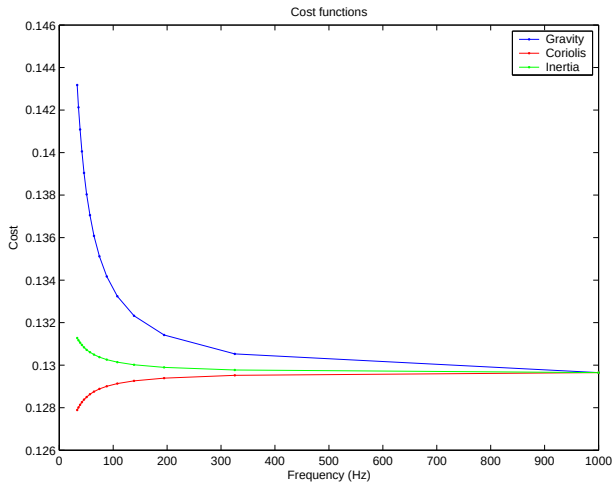


Figure 4. Cost functions

Based on this cost functions it appears that gravity compensation is the more important task therefore we have to allocate it more resources. Costs of Coriolis and inertia compensation are quite similar thus gravity compensation resources allocation is chosen to be twice of Coriolis or inertia ones.

In this application, the period of the feedback scheduler has been fixed to $30ms$ to be larger than the robot control tasks (which limits have been fixed here from $0.5ms$ to $30ms$).

5.3. Feedback scheduling design

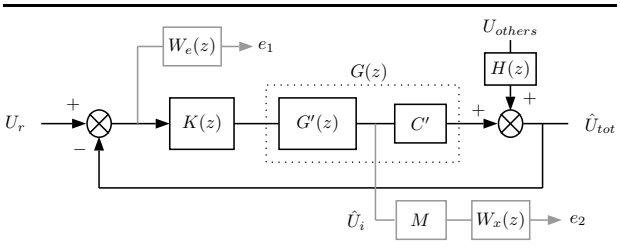


Figure 5. H_∞ design bloc diagram

The bloc diagram of figure (5) is considered for the H_∞ design where $G'(z)$ is the model of the scheduler, the output of which is the vector of all task loads. To get the sum of all task loads, we have $C' = [1\ 1\ 1]$. $H(z)$ represents the sensor dynamical behaviour which measures the load of the other tasks. It may be a first order filter. The template W_e specifies the performances on the load tracking error as follows :

$$W_e(s) = \frac{s/M_s + \omega_b}{s + \omega_s \epsilon} \tag{7}$$

with $M_s = 2$, $\omega_s = 10\, rad/s$, $\epsilon = 0.01$ to obtain a closed-loop settling time of $300\, ms$, a static error less than $1\,\%$ and a good robustness margin. Matrix $M = [1\ -1\ -1]$ and template W_x allow to specify the load allocation between the control tasks. With a large gain in W_x, it leads to:

$$U_{gravity} \approx U_{Coriolis} + U_{inertia},$$

i.e. we allocate more resources for the gravity compensation.

All templates are discretized with a sampling period of 30 ms. Finally discrete-time H_∞ synthesis algorithm produces a discrete-time scheduling controller of order 4.

5.4. Simulations

Simulations are performed using the TrueTime toolbox presented in [9]

Benchmark: The trajectory to be tracked consists in a point to point motion, coming from the position $[-\pi/2, -\pi/2, -\pi/2, -\pi/2, -\pi/2, -\pi/2, -\pi/2]$ to $[\pi/2, 0, \pi/2, 0, \pi/2, 0, \pi/2]$ in the joint space at a constant velocity for each joint. The trajectory duration is set to 2 secs, thus it is slow enough to avoid reaching the actuators limits. The system is observed on a total duration of 2.5 seconds.

Concerning feedback scheduling, the nominal set point of the resource utilisation is 60%. At $t = 1.5s$, a disruptive task appears that needs 30% of the processor load.

Result analysis : From Fig. 6 (periods) the disruptive task produces a transient increase of the processor load at $t = 1.5s$. To reject this disturbance, the scheduling controller increases the periods of the three compensators. On Fig. 6 (loads) it appear that the gravity compensation load is twice those of the inertia or Coriolis load as specified through the M and W_x templates.

On Fig. 6 (Angle) we may check that the control load variations have no noticeable effects on the trajectory (as expected due to the co-design and feedback scheduling strategies), whereas the commands appear to be a little more noisy on Fig. 6 (torques).

Remark 1 *As proposed in [14], the internal process controller can be also designed to take into account timing uncertainties, e.g. the control delays due to preemp-*

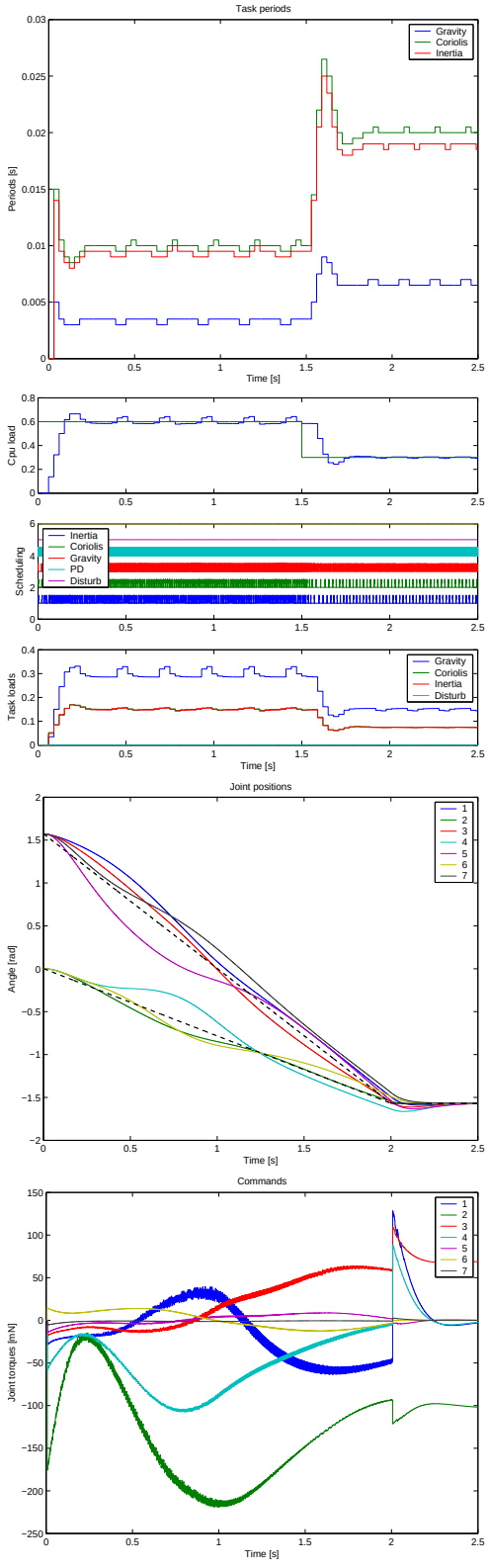


Figure 6. Simulations with TrueTime

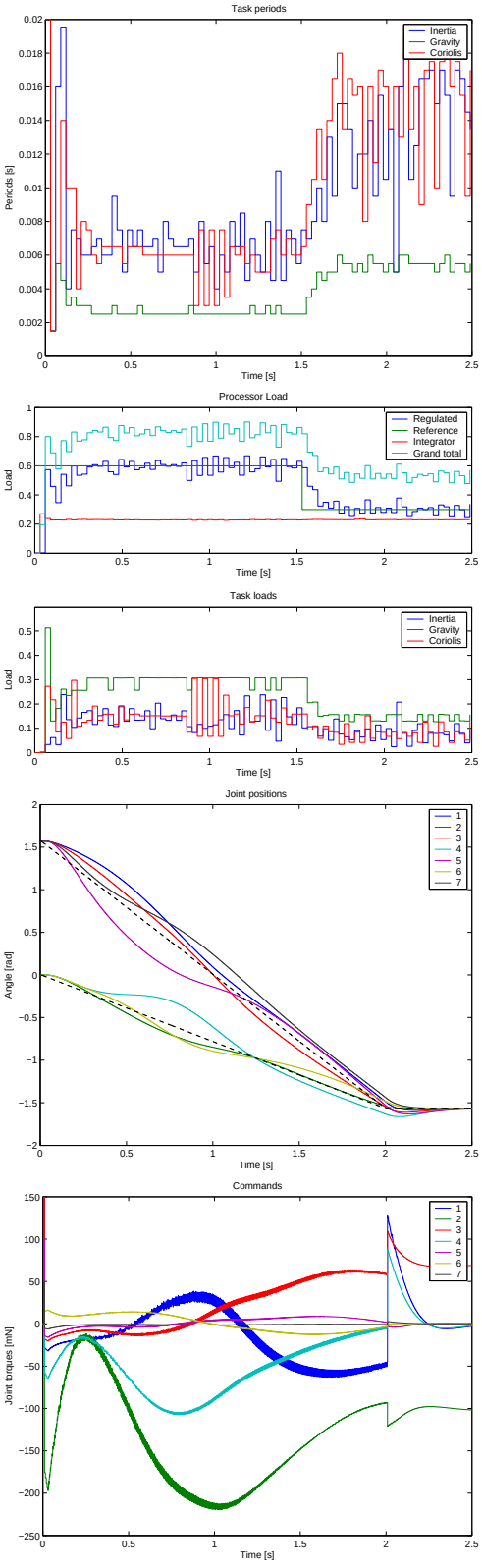


Figure 7. Experiments with RTAI

tions which are unavoidable in real-time control and difficult to accurately predict in a dynamic environment.

6. Feasibility of implementation for feedback schedulers

While Truetime is very useful for fast prototyping and evaluation of scheduling controllers, it remains a simulation tool where the hardware and operating systems are abstracted. Execution times for control tasks are given as arbitrary values and execution times for the scheduler and system level housekeeping tasks are neglected. Therefore to assess the feasibility and practical interest of such adaptive closed-loop scheduling we have developed a feedback scheduler prototype on top of an existing robot controller. The current prototype runs under RTAI, the Realtime Linux Application Interface for Linux (`http://www.aero.polimi.it/~rtai/`). This RTOS provides an efficient real-time scheduler (with a recorded interrupt latency range of $1 - -7\mu sec$ with the Pentium II 450 MHz used for this setup) and allows time-stamping of events with the resolution of the built-in timer, thus the response times of each module can be accurately measured and plotted.

The original controller uses the so-called "Computing Torque Controller" which is split into several computing modules to implement a multi-rate controller as in [15] (Figure 8):

- CompTorque *is* the controller, updating the control torque vector U from the state (position and velocity) error vector $Q_d - Q$ using a PD algorithm which gains are tuned to provide a comfortable stability margin. The sampling rate and latency of this block have a strong impact on the system's performance and stability. The extra inputs coming from the robot model's computations are used to provide feed-forward; in the previous simulation framework, it is called the PD control task. As well, the period of this task is constant for stability requirements.
- Gravity, Coriolis and Inertia compute an explicit model of the robot arm's dynamics. They are costly to compute but it has been shown through simulations that they can be executed several times slower than the main controller with a moderate impact on the control performance and stability;
- GeneTraj provides the desired trajectory via-points at a fixed rate;
- The behaviour of the periodic control modules is supervised by a event driven reactive task in charge of setting up the system and exception handling, e.g. initialising the real-time tasks and cleanly stopping the system when it reaches the nominal termination state or when the control error exceeds a predefined threshold.

Control modules with different sampling rates communicate through asynchronous protected shared memories ; they are infinite loops triggered by clocks derived from the built-in timer by a middleware clock generator task (ClockGen).

As our goal is testing the feasibility of feedback-scheduling with no modifications of the internal of the operating system, the scheduling feedback loop is implemented as follows:

- The feedback scheduler is implemented as an additional real-time periodic task, i.e. a control module which function is specified and encoded by the control designer. The inputs are the measured response times of the control tasks. The set point is a desired global computing load. Outputs are the sampling periods of the Gravity, Coriolis and Inertia control tasks;
- The period of CompTorque is fixed in this particular experiment. It runs at a fixed period of 1ms so that the stability of the system can be preserved. It implements the algorithm described in section 5.1 and reads the last available outputs of Gravity, Coriolis and Inertia via protected shared memories;
- The scheduling regulator adapts the periods of the Gravity, Coriolis and Inertia compensation modules and their computing load must be evaluated. The evaluation of the computing load is more or less easy and precise according the features of the operating system in use.

  On one hand, the direct measure of tasks execution times is out of the scope of the API of a POSIX system. In that case, the computing load could be evaluated via the *time of response* of the tasks, but as the preemption phases cannot be discarded the global load would be over-estimated.

  On the other hand, the RTAI kernel keeps track of the *rt_tasks* execution cycles, thus allowing to recover a precise measure of the control tasks execution times from the scheduling regulator.

  Deadlines misses are checked and reported to the supervisor ; they are not taken into account in this preliminary setup but may be used in future improvements of a load estimator, for anti-windup and for overload handling;
- In this experiment the robot is still (cautiously) simulated: the drivers call a numerical integrator running the robot's dynamics model every time a

measure is taken or a new control vector is sent by the CompTorque task. Therefore the controller and simulated process time scales are coherent, but the simulation adds unpredicted latencies in the loop. The corresponding measured added load is about 25% of the Pentium CPU used for the setup which is fortunately strongly over-sized for this robot controller;

- Priorities are ordered according to the relative urgency and weight of these function blocks on the system's behaviour: $ClockGen \succ Supervisor \succ FedSched \succ CompTorque \succ GeneTraj \succ Gravity \succ Coriolis \succ Inertia$
- The initial periods has been set to 20 msec for the three tasks with a variable and controlled period, i.e. Gravity, Coriolis and Inertia. The PD control task is run at a fixed rate of 1ms and the trajectory generator provides new set points every 5ms. The feedback scheduler is run every 30 msec.
- The sampling frequency of the clock generator is set to 2 KHz thus allowing to increment or decrement the control tasks clocks by $500\mu sec$ steps. Observing and managing the system at a faster rate would induce a very high system's load due to too many context switches.
- The desired trajectory is the same as in 5. Preliminary experiments shown that the robot's numerical integration spends about 25% of the available CPU power. As we need some load margin to avoid transient overruns the desired load is initially set to 0.6. At time 1.5 sec it is decreased to 0.3 to make room for a disturbing incoming task proposed for admission.

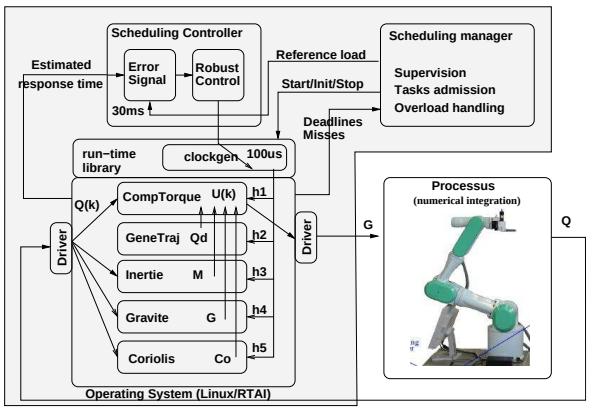


Figure 8. Feed-back scheduling experiment

As plot in Figure 7 the very first experimental results are encouraging: they show that such a feedback scheduling architecture can be quite easily designed and implemented on top of an off-the-shelf real-time operating system with fixed priority and preemption. Examining the plots and the kernel reporting file calls for the following comments:

- As the feedback scheduler is a simple feedback algorithm running at a slow rate its computing cost is quite low (about $75\mu sec$ every $30ms$). On the other hand observing and managing the system at a high rate can be very costly due to numerous context switches. A more efficient implementation is currently studied to decrease the system's cost.
- The tasks loads can be directly measured using a specific, non portable feature of the RTAI kernel. As response times are easier to measure (e.g.from a POSIX API), using an execution time estimator would increase the portability of the system. Designing such a low-cost and reliable estimator remains to be done.
- Overshoots in the control periods and load lead to transient overload and deadline misses. These events are not currently processed leading to an avalanche of timing faults and to a system unstability and failure. Overruns and other timing errors must be adequately processed at the supervision level to more safely manage the system and make use of the full range of available processing power.

7. Conclusion

In this paper, a methodology for control and scheduling co-design is proposed. Besides the usual modelling errors in the process model a digital closed-loop controller is also submitted to timing disturbances coming from the implementation: they are difficult to be accurately predicted and studying the impact of timing deviations in feedback loops is still a largely unexplored domain. Hence a natural idea consists in using robust control theory to design controllers to be weakly timing sensitive. Besides process control this idea can be used also to design a feedback scheduling loop to implement robust on-line adaption of the scheduling parameters according to estimates of the computing activity. Thus this resource allocation control loop is used to fulfil the plant control objective under constraint of limited computing resource.

An integrated control-scheduling framework is proposed. The control periods are weighted according to their impact on the control performance. An outer scheduling controller then regulates in real-time the

CPU load according to the allocated computing power. Indeed the control synthesis of the feedback scheduler has been provided using the H_∞ control theory, and the gain of the controller is adapted on-line using the estimated execution times of the control tasks. In all cases the effectiveness of the controllers lies in a right choice of the design weighting functions used to specify the trade-off between concurrent constraints. Therefore the designer(s) must have knowledge on both the process and computing platform capabilities to be able to best fit the end-user requirements.

Some simulation and experiment results have been given, which emphasises the interest of this approach and a software prototype has been designed to assess the feasibility of the approach using an off-the-shelf real-time operating system.

Further works concern the improvement of robust control schemes for both the process and scheduler controllers. A better insight in control w.r.t. timing uncertainties will be necessary to efficiently shape the weighting between control and computing constraints. Implementation feasibility must be taken into account and the system's supervision must be improved. Finally choosing strategies and tuning parameters leading to an effective trade-off which fit with the end-user's requirements needs a common understanding and cooperation between control and computer scientists and engineers.

References

[1] K. Åström and B. Wittenmark. *Computer-Controlled Systems*. Information and systems sciences series. Prentice Hall, New Jersey, 3rd edition, 1997.

[2] G. Bernat, A. Burns, and A. Llamosí. Weakly hard real-time systems. *IEEE Transactions on Computers*, 50(4):308–321, 2001.

[3] G. Buttazzo and L. Abeni. Adaptive rate control through elastic scheduling. In *39th Conference on Decision and Control*, Sydney, Australia, 2000.

[4] A. Cervin. *Integrated Control and Real-Time Scheduling*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden, Apr. 2003.

[5] A. Cervin and J. Eker. Feedback scheduling of contol tasks. In *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, Dec. 2000.

[6] A. Cervin, J. Eker, B. Bernhardsson, and K.-E. Årzén. Feedback-feedforward scheduling of control tasks. *Real-Time Systems*, 23(1), 2002.

[7] J. Eker, P. Hagander, and K.-E. Årzén. A feedback scheduler for real-time control tasks. *Control Engineering Practice*, 8(12):1369–1378, 2000.

[8] J. Eker, P. Hagander, and K.-E. Årzén. A feedback scheduler for real-time controller tasks. *Control Engineering Practice*, 8(12):pp 1369–1378, 2000.

[9] D. Henriksson, A. Cervin, and K.-E. Årzén. Truetime: Simulation of control loops under shared computer resources. In *15th IFAC World Congress on Automatic Control*, Barcelona, Spain, July 2002.

[10] C. Lu, J.-A. Stankovic, G. Tao, and S.-H. Son. Feedback control real-time scheduling: Framework, modeling, and algorithms. *Real Time Systems*, 23(1):85–126, 2002.

[11] M. Ryu, S. Hong, and M. Saksena. Streamlining real-time controller design: from performance specifications to end-to-end timing constraints. In *IEEE Real Time Systems Symposium*, 1997.

[12] K. Sandström and C. Norström. Managing complex temporal requirements in real-time control systems. In *9th IEEE Int. Conf. and Workshop on the Engineering of Computer-Based Systems (ECBS'02)*, Lund, Sweden, 2002.

[13] M. Schinkel, W.-H. Chen, and A. Rantzer. Optimal control for systems with varying sampling rate. In *Proceedings of American Control Conference*, Anchorage, May 2002.

[14] O. Sename, D. Simon, and D. Robert. Feedback scheduling for real-time control of systems with communication delays. In *ETFA'03 9th IEEE International Conference on Emerging Technologies and Factory Automation*, Lisbonne, september 2003.

[15] D. Simon, E. Castillo, and P. Freedman. Design and analysis of synchronization for real-time closed-loop control in robotics. *IEEE Trans. on Control Systems Technology*, 6(4):445–461, july 1998.

[16] D. Simon, K. Kapellos, and B. Espiau. Control laws, tasks and procedures with orccad: Application to the control of an underwater arm. *Int. Journal of Systems Science*, 29(10):1081–1098, 1998.

[17] D. Simon, O. Sename, D. Robert, and O. Testa. Real-time and delay-dependent control co-design through feedback scheduling. In *CERTS'03 Workshop on Co-design in Embedded Real-time Systems*, Porto, july 2003. ECRTS.

[18] M. Törngren. Fundamentals of implementing real-time control applications in distributed computer systems. *Real Time Systems*, 14(3):219–250, 1998.

[19] K. Zhou, J. C. Doyle, and K. Glover. *Robust and optimal control*. Prentice-Hall Inc., 1996.

Bibliographie

Abeni, L. and Buttazzo, G. (1998). Integrating multimedia applications in hard real-time systems. In *Proceedings of the 19th Real-Time Systems Symposium*, Madrid, Spain.

Abeni, L., Palopoli, L., Lipari, G., and Walpole, J. (2002). Analysis of a reservation-based feedback scheduler. In *Proceedings of the 23rd Real-Time Systems Symposium*, Austin, Texas.

Albertos, P. and Crespo, A. (1999). Real-time control of non-uniformly sampled systems. *Control Engineering Practice*, 7(4) :445–458.

Albertos, P., Vallés, M., and Valera, A. (2003). Controller transfer under sampling rate dynamic changes. In *Proceedings of the European Control Conference 2003*, Cambridge, UK.

Apkarian, P. and Adams, R. J. (1998). Advanced gain-scheduling techniques for uncertain systems. *IEEE Transactions on Control Systems Technology*, 6(1) :21–32.

Apkarian, P., Gahinet, P., and Becker, G. (1995). Self-scheduled h(infinity) control of linear parameter-varying systems : A design example. *Automatica*, 31(9) :1251–1262.

Åström, K. J. and Wittenmark, B. (1997). *Computer-Controlled Systems*. Information and systems sciences series. Prentice Hall, New Jersey, 3rd edition.

Balbastre, P., Ripoll, I., Vidal, J., and Crespo, A. (2004). A task model to reduce control delays. *Real-Time Systems*, 27(3) :215–236.

Becker, G. and Packard, A. (1994). Robust performance of linear parametrically varying systems using parametrically-dependent linear feedback. *Systems and Control Letters*, 23(3) :205–216.

Bernussou, J., Geromel, J., and Peres, P. (1989). A linear programming oriented procedure for quadratic stabilization of uncertain systems. *Systems and Control Letters*, 13(1) :65–72.

Bini, E. and Natale, M. D. (2005). Optimal task rate selection in fixed priority systems. In *Proceedings of the 26th IEEE Real-Time Systems Symposium*, Miami, FL.

Boyd, S., ElGhaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear Matrix Inequalities in systems and Control*, volume 15 of *SIAM Studies in Applied Mathematics*. SIAM.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Burns, A. and Wellings, A. (2001). *Real-Time Systems and Programming Languages*. Addison-Wesley.

Buttazzo, G., Lipari, G., and Abeni, L. (1998). Elastic task model for adaptive rate control. In *Proceeding of the IEEE Real-Time Systems Symposium*, Madrid, Spain.

Caccamo, M., Buttazzo, G., and Sha, L. (2000). Elastic feedback control. In *Proceedings of the 12th Euromicro Conference on Real-Time Systems*, Stockholm, Sweden.

Cervin, A. (1999). Improved scheduling of control tasks. In *Proceedings of the 11th Euromicro Conference on Real-Time Systems*, pages 4–10, York, UK.

Cervin, A. (2003). *Integrated Control and Real-Time Scheduling*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.

Cervin, A. and Eker, J. (2005). Control-scheduling codesign of real-time systems : The control server approach. *Journal of Embedded Computing*.

Cervin, A., Eker, J., Bernhardsson, B., and Årzén, K.-E. (2002). Feedback-feedforward scheduling of control tasks. *Real-Time Systems*, 23(1–2) :25–53.

Cervin, A., Lincoln, B., Eker, J., Årzén, K.-E., and Buttazzo, G. (2004). The jitter margin and its application in the design of real-time control systems. In *Proc. 10th International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA)*, Göteborg, Sweden.

Chen, T. and Francis, B. (1995). *Optimal Sampled-Data Control Systems*. Springer.

de Oliveira, M., Bernussou, J., and Geromel, J. (1999). A new discrete-time robust stability condition. *Systems & Control Letters*, 37(4) :261–265.

Eker, J., Hagander, P., and Årzén, K.-E. (2000). A feedback scheduler for real-time controller tasks. *Control Engineering Practice*, 8(12) :1369–1378.

Gahinet, P. (1996). Explicit controller formulas for lmi-based h_∞ synthesis. *Automatica*, 32(7) :1007–14.

Gahinet, P. and Apkarian, P. (1994). A linear matrix inequality approach to h_∞ control. *International Journal of Robust and Nonlinear Control*, 4 :421–448.

Gahinet, P., Nemirovski, A., Laub, A., and Chilali, M. (1996). *LMI Control Toolbox*.

Henriksson, D. and Cervin, A. (2005). Optimal on-line sampling period assignment for real-time control tasks based on plant state information. In *Proceeding of the 44th IEEE Conference on Decision and Control*.

Henriksson, D., Cervin, A., and Årzén, K.-E. (2002). TrueTime : Simulation of control loops under shared computer resources. In *Proceedings of the 15th IFAC World Congress on Automatic Control*, Barcelona, Spain.

Hu, B. and Michel, A. N. (2000). Stability analysis of digital feedback control systems with time-varying sampling periods. *Automatica*, 36(6) :897–906.

Joseph, M. and Pandya, P. (1986). Finding response time in a real-time system. *The Computer Journal*, 29(5) :390–395.

Kao, C.-Y. and Lincoln, B. (2004). Simple stability criteria for systems with time-varying delays. *Automatica*.

Leung, J. Y. T. and Whitehead, J. (1982). On the complexity of fixed priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2(4) :237–250.

Lincoln, B. (2002). Jitter compensation in digital control systems. In *Proceedings of the 2002 American Control Conference*.

Lincoln, B. and Bernhardsson, B. (2000). Efficient pruning of search trees in lqr control of switched linear systems. In *Proceedings of the 39th IEEE Conference on Decision and Control*.

Lincoln, B. and Cervin, A. (2002). Jitterbug : A tool for analysis of real-time control performance. In *Proceedings of the 41st IEEE Conference on Decision and Control*, Las Vegas, NV.

Liu, C. L. and Layland, J. W. (1973). Scheduling algorithms for multi-programming in a hard-real-time environnement. *Journal of the ACM*, 20(1) :46–61.

Lu, C., Stankovic, J. A., Abdelzaher, T. F., Tao, G., Son, S. H., and Marley, M. (2000). Performance specifications and metrics for adaptive real-time systems. In *21th IEEE Real-Time Systems Symposium (RTSS 2000)*, Orlando, FL.

Lu, C., Stankovic, J. A., Son, S. H., and Tao, G. (2002). Feedback control real-time scheduling : Framework, modeling and algorithms. *Real-Time Systems Journal, Special Issue on Control-Theoretical Approaches to Real-Time Computing*, 23(1/2) :85–126.

Martí, P., Fuertes, J. M., Fohler, G., and Ramamritham, K. (2001). Jitter compensation for real-time control systems. In *Proceedings of the 22nd IEEE Real-Time Systems Symposium (RTSS 2001)*, London, UK.

Martí, P., Lin, C., Brandt, S. A., Velasco, M., and Fuertes, J. M. (2004). Optimal state feedback based ressource allocation for ressource-constrained control tasks. In *Proceedings of the 25th IEEE Real-Time Systems Symposium*, Lisbon, Portugal.

Natale, O. R., Sename, O., and de Wit, C. C. (2004). Inverted pendulum stabilization through the ethernet network, performance analysis. In *Proceedings of the 2004 American Control Conference*.

Nilsson, J. (1998). *Real-Time Control Systems with Delays*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.

Oliveira, M., Geromel, J., and Bernussou, J. (2002). Extended h2 and hinf norm characterizations and controller parametrizations for discrete-time systems. *International Journal of Control*, 75(9) :666–679.

Packard, A. (1994). Gain scheduling via linear fractional transformations. *Systems and Control Letters*, 22(2).

Palopoli, L., Abeni, L., and Lipari, G. (2003). On the application of hybrid control to cpu reservations. In Maler, O. and Pnueli, A., editors, *Hybrid Systems : Computation and Control*, Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, Germany.

Palopoli, L., Pinello, C., Sangiovanni-Vincentelli, A., Elghaoui, L., and Bicchi, A. (2002). Synthesis of robust control systems under resource constraints. In *Hybrid Systems : Computation and Control*, Standford.

Rehbinder, H. and Sanfridson, M. (2000). Integration of off-line scheduling and optimal control. In *Proceedings 12th Euromicro Conference on Real-Time Systems (Euromicro RTS 2000)*.

Richard, J. (2003). Time-delay systems : an overview of some recent advances and open problems. *Automatica*, 39 :1667–1694.

Robert, D., Sename, O., and Simon, D. (2005). Sampling period dependent rst controller used in control/scheduling co-design. In *Proceedings of the 16 th IFAC World Congress*, Czech Republic.

Ryu, M. and Hong, S. (1998). Toward automatic synthesis of schedulable real-time controllers. *International Journal of Integrated Computer-Aided Engineering*, 5(3) :261–277.

Ryu, M., Hong, S., and Saksena, M. (1997). Streamlining real-time controller design : From performance specifications to end-to-end timing constraints. In *Proceedings. 3rd IEEE Real-Time Technology and Applications Symposium*, pages 91–99.

Sala, A. (2005). Computer control under time-varying sampling period :an lmi gridding approach. *Automatica*, 41(12) :2077–2082.

Scherer, C., Gahinet, P., and Mahmoud, C. (1997). Multiobjective output-feedback control via lmi optimization. *IEEE Transactions on Automatic Control*, 42(7) :896–911.

Scherer, C. W. (2001). Lpv control and full block multipliers. *Automatica*, 37(3) :361–375.

Scorletti, G. and El Ghaoui, L. (1998). Improved lmi conditions for gain-scheduling and related control problems. *International Journal of Robust and Nonlinear Control*, 8(1) :845–877.

Sename, O., Simon, D., and Robert, D. (2003). Feedback scheduling for real-time control of systems with communication delays. In *Proceedings of the 2003 IEEE Conference on Emerging Technologies and Factory Automation*, pages 454–61, Lisbon, Portugal.

Seto, D., Lehoczky, J. P., Sha, L., and Shin, K. G. (1996). On task schedulability in real-time control systems. In *Proceedings of the 17th IEEE RealTime Systems Symposium*, pages 13–21, Washington, DC.

Sha, L., T., A., Arzen, K., Cervin, A., Baker, T., Burns, A., Buttazzo, G., Caccamo, M., Lehoczky, J., and Mok, A. (2004). Real time scheduling theory : A historical perspective. *Real-Time Systems*, 28(2-3) :101–155.

Shamma, J. F. and Athans, M. (1991). Guaranteed properties of gain scheduled control for linear parameter-varying plants. *Automatica*, 27 :559–564.

Shin, K. G. and Meissner, C. L. (1999). Adaptation of control system performance by task reallocation and period modification. In *Proceedings of 11th Euromicro Conf. on Real-Time Systems*, pages 29–36, York, UK.

Simon, D., Robert, D., and Sename, O. (2005). Robust control/scheduling co-design : application to robot control. In *Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium*, San Francisco, United States.

Skogestad, S. and Postlethwaite, I. (1996). *Multivariable Feedback Control : analysis and design*. John Wiley and Sons.

Stankovic, J. A., Lu, C., Son, S. H., and Tao, G. (1999). The case for feedback control real-time scheduling. In *Proceedings of the 11th Euromicro Conference on RealTime Systems*, pages 11–20, York, England.

Stankovic, J. A., Spuri, M., K., R., and C., B. G. (1998). *Deadline Scheduling for Real-Time Systems - EDF and Related Algorithms*. Kluwer Academic publishers.

Witrant, E. (2005). *Stabilisation des systèmes commandés par réseaux*. PhD thesis, Institut National Polytechnique de Grenoble.

Wu, F. (2001). A generalized lpv system analysis and control synthesis framework. *International Journal of Control*, 74(7) :745.

Zhao, Q. C. and Zheng, D. Z. (1999). Stable and real-time scheduling of a class of hybrid dynamic systems. *Discrete Event Dynamic Systems : Theory and Applications*, 9 :45–64.

Zhou, K., Doyle, J., and Glover, K. (1996). *Robust and Optimal Control*. Prentice Hall.

Résumé : Lors du développement d'un système contrôlé par ordinateur, il y a traditionnellement une séparation forte entre les conceptions de la loi de commande et de l'ordonnancement des tâches du calculateur. Dans les systèmes embarqués, où les ressources sont limitées et variables, cette séparation rigide conduit à une sous utilisation du processeur. Pour améliorer l'interaction commande/ordonnancement, nous proposons d'adapter les ressources nécessaires au calcul des lois de commande en agissant sur leur période. Nous présentons tout d'abord une structure d'ordonnancement régulé. La contribution principale concerne ensuite le développement d'une méthodologie de synthèse d'une loi de commande à période d'échantillonnage variable basée sur une approche de type Hinfini dédiée aux systèmes linéaires à paramètres variants. Les performances de cette loi de commande ainsi que son interaction avec l'ordonnancement sont illustrées en simulation et sur un procédé expérimental.

Mots-clés : loi de commande à période variable, commande Hinf, commande LPV, ordonnancement temps réel flexible

Abstract : During the design of a computer controlled system, there is a traditional separation of concern between control law and scheduling design. In embedded systems, with limited and varying resources, this separation leads to under utilization of the processor. To improve the control/scheduling interactions we propose to adapt the resources usage of the control law computations by adjusting their sampling period. Some preliminary results concern the design of a feedback scheduling using a robust control approach. The main result of this work is the development of a control synthesis methodology with variable sampling period using the Hinfinity approach for linear varying parameter systems. The performance of the control law and its interaction with the scheduling are illustrated in simulation and by experimental results.

Keywords : variable sampling control, Hinf control, LPV control, flexible scheduling