Projet Robot LEGO : Cahier des charges Cours Modélisation Réaliste et Implantation Multi-tâche

Thao Dang et Pascal Raymond

15 novembre 2016

1 Modélisation réaliste et commande robuste

1. **Problème 1.a** - Modéliser les sources possibles d'imprécision et perturbations dans le robot et des contrôleurs en pratique. Par simulation, étudier leur influence sur la stabilité du système global.
2. **Problème 1.b** - Etudier la robustesse des contrôleur d'angle en utilisant des critère de stabilité pour des systèmes incertains et avec retards. Justifier le choix des paramètres : la période T d'échantillonnage, ki_θ (de l'action proportionielle) et kp_θ (de l'action intégrale) des contrôleurs (voir TD1).
3. **Problème 1.c** - Le délai τ dans la réaction moteur est introduit dans la fonction transfert. Etudier la stabilité du système en fonction de τ (voir TD2).

2 Commande multi-objective - suiveur de ligne et anti-collision

Problème 2 - Créer en Simulink un système de commande de robots LEGO qui réalise les 2 fonctions suivantes :

Régulateur de trajectoires. Cette fonction assure que le robot suit une trajectoire désirée tout en évitant des obstacles. Elle envoie périodiquement des valeurs de vitesse des deux roues aux moteurs. Ces valeurs sont calculées par deux contrôleurs (de distance et d'angle) en fonction des informations courantes sur la position et l'orientation x, y, θ.

Planification et anti-collision.

– Anti-collision. La *stratégie d'anti-collision* que l'on va utiliser est la suivante.

1. D'abord, le robot s'arrête (notons que le robot prend un certain temps pour s'arrêter complètement !).
2. Ensuite, le robot fait un demi-tour sur place. Supposons que le robot tourne vers le capteur à gauche, pour détecter le moment où le robot termine le demi-tour, on peut détecter une séquence de valeurs "Blanc-Noir-Blanc" dans la sortie du capteur à gauche.

– Planification

1. Dans le mode suiveur de ligne, le planificateur détermine d'une manière continue les valeurs de consigne ϵ_d et ϵ_θ et puis communique ces valeurs aux contrôleurs (de distance et d'angle) qui calculent les vitesses des deux roues.
2. Pour l'anti-collision, il est important que le robot s'arrê le plus rapidement possible, *il est donc plus efficace pour le planificateur de spécifier directement les vitesses v_d et v_g des deux roues*, au lieu de spécifier les erreurs ϵ_d et ϵ_θ à corriger.

Détecter une séquence de couleurs. Pour détecter une séquence de valeurs "Blanc-Noir-Blanc", il faut mémoriser, par exemple, l'événement qu'un

capteur voit la couleur "Blanc". Pour ceci, on peut ulitiser un bloc "Logical Or" dont une entrée est connectée à la sortie d'un comparateur (voir la Figure 1 où N_b est le seuil de la couleur blanche). Après terminer le demi-tour,

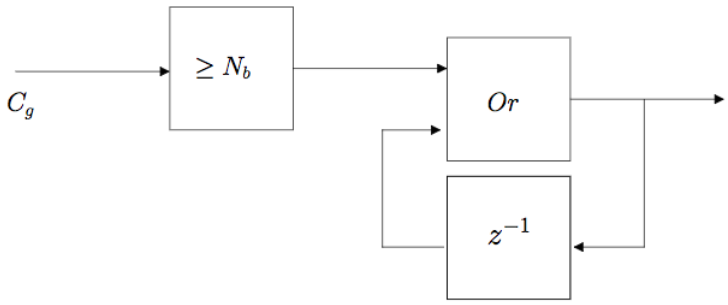

FIGURE 1 – Mémorisation de l'événement de détection de couleur blanche.

le contrôleur doir revenir dans l'état avant le mode d'anti-collision, il est utile de pouvoir "réinitialiser" les variables importantes du contrôleur. Pour ceci, voir la Figure 2 pour un exemple de bloc "mémoire avec reset".

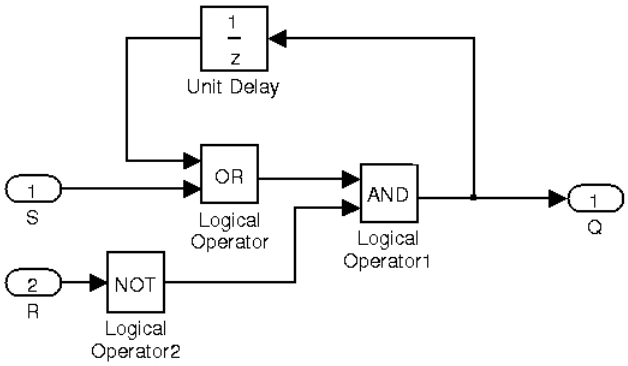

FIGURE 2 – Mémoire avec reset.

3 Implantation multi-tâche

Problème 3 - Utiliser la chaîne **mdl2lus2osek** pour générer le code du contrôleur et l'éxecuter sur les robots en utilisant deux méthodes : mono-

tâche et multi-tâche.

1. Pour l'implantation multi-tâche, séparer le contrôleur en deux sous-systèmes représentant deux tâches avec différentes périodes , puis générer le code Lustre (en utilisant l'outil **mdl2lus**) et puis le code C (en utilisant le compilateur **lus2c**) pour chaque sous-système. Expliquer la séparation.
2. Ecrire le code "de glue" en utilisant **nxtOSEK** (voir l'API de nxtOSEK `http://lejos-osek.sourceforge.net/ecrobot_c_api.htm` pour les fonctions de manipulation de capteur et de moteur). Le code de glue doit contenir dans la phase d'initialisation (par exemple dans la fonction "`usr_init`") une étape de "calibrage" de capteur et de moteur.
3. Faire des expériments avec le robot et ajuster la commande si besoin en modifiant les modèles SIMULINK. Expliquer la procédure et interpréter le résultat.