Cours ISC - 2A
TP Commande d'un robot suiveur de ligne

Thao Dang
CNRS-VERIMAG, 2, av. de Vignate, 38610 Gieres, France

29 février 2012

1 Suivre une ligne noire

Nous allons maintenant considérer le problème de faire suivre au robot une ligne en utilisant des capteurs de lumière, positionnés sur le devant et qui pointent vers le sol afin de détecter une ligne noire. Le robot possède deux moteurs connectés à ses deux roues. On doit connaître les valeurs que les capteurs retournent quand ils voient une « couleur » entre le le noir et le blanc. Autour de chaque capteur, il y a une petite zone qu'il peut voir.

Hypothèse : Supposons que les capteurs soient calibrés sur une échelle de 0 (noir) à 100 (blanc) (**il faut vérifier cette hypothèse et ajuster le calibrage si besoin**). Si la zone est complètement noire, le capteur retourne 0. Si la zone est complètement blanche, le capteur retourne 100.

Si un capteur retourne une valeur $c < 100$, on sait que le robot n'est plus aligné avec la ligne et il faut le faire tourner vers ce capteur d'un angle proportionel à cette valeur c.

Un exemple d'utilisation des informations fournies par les capteurs est comme suit (on suppose que le robot est initialement aligné sur la ligne) :
- Si les deux capteurs voient le blanc, le robot va tout droit (sauf dans des cas particuliers tels que dans une manœuvre de demi-tour).
- Si un seul capteur voit le blanc, le robot tourne vers le capteur qui voit le noir.

– Si les deux capteurs voient le noir, le robot s'arrête et attend l'intervention de l'utilisateur pour le remettre sur la ligne.

Il faut raffiner/ajuster cette stratégie en fonction de la ligne réelle.

Pour réaliser cette stratégie, notons que l'orientation θ du robot est contrôlée par la différence entre les vitesses des deux roues v_g et v_d, ce qui est réalisé par le contrôleur d'angle. La vitesse du robot est contrôlée par la moyenne de v_g et v_d, ce qui est réalisé par le contrôleur de distance.

Dans le TD précédent, nous avons calculé deux contrôleurs qui réalisent les trajectoires du robot en fonction de la différence entre les coordonnées courantes du robot et celles du point cible (x_{final}, y_{final}). Nous allons maintenant combiner ces contrôleurs avec un planificateur de trajectoire qui change le point cible « implicitement » en fonction des informations venant des capteurs.

Le problème est maintenant de suivre une ligne que le robot ne peut pas connaître. Il peut seulement, à partir des informations venant des capteurs, estimer sa déviation par rapport à la ligne.

Nous allons d'abord contrôler l'orientation du robot, c'est-à-dire déterminer la commande qui agit sur la différence $v_g - v_d$. Pour corriger l'angle du robot, on définit l'erreur d'angle (la différence entre l'angle courant et l'angle désiré) que l'on veut ramener à 0. Notons que dans le contexte du suiveur de ligne, on peut estimer cette différence sans connaître l'angle désiré.

Ensuite, cette erreur est donnée en entrée du PID du contrôleur d'angle.

On doit encore contrôler la vitesse du robot, car le contrôleur d'angle détermine seulement la différence entre v_g et v_d.

– Quand le robot ne doit pas tourner, on peut laisser la vitesse du robot constante, par exemple en créant à l'entrée du contrôleur de distance une « erreur »de distance constante. Dans ce cas, une solution plus simple est de désactiver la partie intégrale du PID et de garder l'entrée constante.
– Quand le robot doit tourner, si la vitesse courante du robot est trop grande et que la ligne est très courbée,le robot peut faire de grands dépassements. Donc, quand le robot doit tourner beaucoup, on devra réduire sa vitesse (par exemple, en réduisant l'erreur à l'entrée du contrôleur de distance).

2 Création d'un modèle Simulink

Un modèle de robot avec simulation des capteurs de lumière se trouve sur la page web d'Alexandre Donzé (`http://www-verimag.imag.fr/~donze`)

- Recupérez l'archive sur la page `http://www-verimag.imag.fr/~donze/enseignements/isc/Simulink.html` et décompressez-la dans un dossier sur votre home.
- Lancez Matlab/Simulink et ouvrez le modèle *Robot_and_Environment.mdl*
- Complétez le bloc Controller, pour cela il vous faudra contruire :

  un bloc planificateur qui calcul l'erreur d'orientation ϵ_θ et de vitesse ϵ_δ en fonction des valeurs retournées par les capteurs de lumière C_d et C_g.

  Deux contrôleurs PID pour l'orientation et la vitesse, (vous pouvez récupérer ceux du précédent TP).
- testez votre modèle en effectuant des simulations (dans Simulink : Simulation>Start ou ctrl-t) après avoir initialisé les paramètres du modèle avec le script *Init_Robot_and_Environment.m*.