« Grains d'Usage robotique »

Asservissement de distance entre plusieurs mBots

Objectif de la manipulation : découvrir ce qu'est un asservissement grâce à un mBot asservi en position (distance) vis-à-vis d'un autre mBot « meneur ». Application aux trains de camion ou aux véhicules autonomes.

Public concerné : lycées (SNT, NSI)

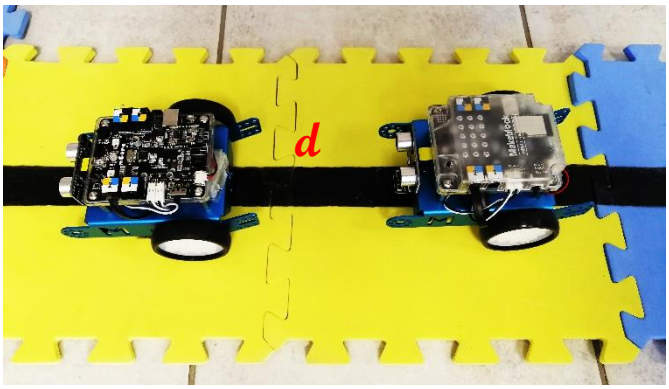
Prérequis : connaissance d'un langage de programmation pour robot mBot (Blocks, Python, C)

Matériel nécessaire : deux (ou plus) mBots, un circuit avec pistes. (Disponibles *au SAMS)*

1- Le défi :

Réaliser un train de véhicules autonomes asservis en distance (***d***) à un véhicule meneur :
- Si le « meneur » accélère ou décélère, le « suiveur » en fait autant de manière à retrouver la distance « d » fixée en consigne.

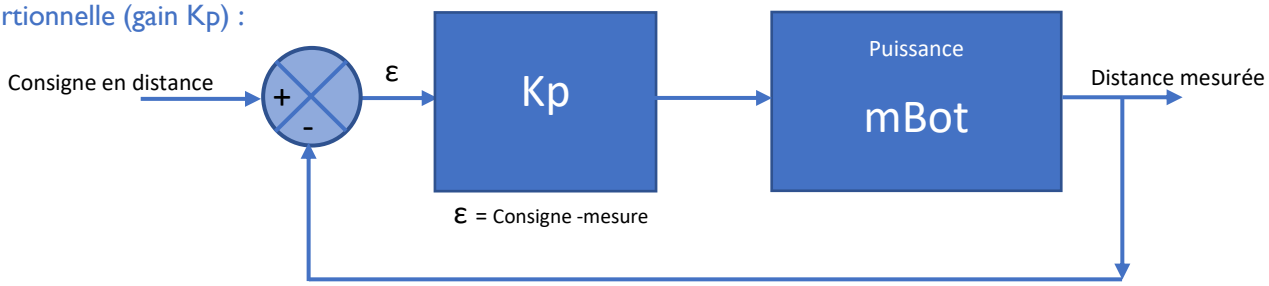
Remarque : par soucis de simplification, les robots sont ici filoguidés et ne peuvent quitter le circuit.



2- Asservissement de distance entre véhicules: (Cycle 4 – Lycées)

Nos robots étant équipés de télémètres à ultra-sons, il s'agira ici d'asservir le robot suiveur à celui meneur en contrôlant la vitesse du robot suiveur sur la base de la distance mesurée. Ainsi la distance entre véhicules sera régulée à la distance de consigne. A noter qu'il n'existe pas de problème de suivi dans les virages, les robots étant filo-guidés sur un circuit.

Asservissement « proportionnel » : Il s'agira ici simplement d'asservir la commande à la distance de façon directe : plus la distance s'écarte de la consigne, plus la commande de vitesse est augmentée, cela de façon proportionnelle (gain Kp) :



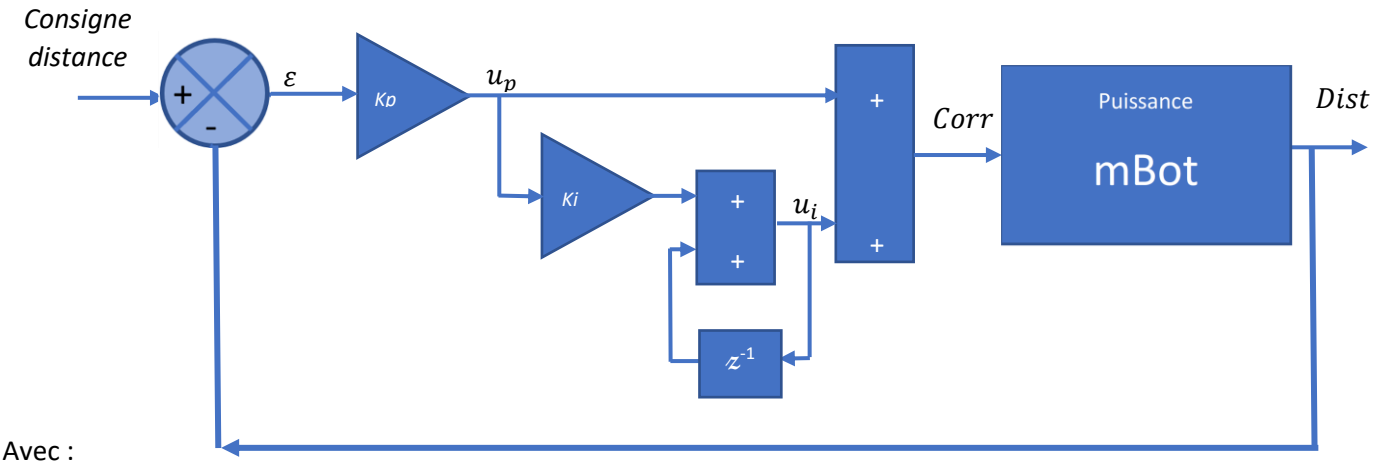
Avec ce type de correction, il existera toujours une « erreur de position » liée au fait que si ε valait 0 (distance mesurée=distance consigne), la consigne serait nulle et le robot serait arrêté (!) il s'ensuivrait que la distance augmenterait…et que l'erreur ne pourrait donc être nulle. En pratique la distance se stabilisera à une distance d'autant plus proche de la consigne que Kp sera grand…mais au prix d'une instabilité croissante.

Afin d'annuler cette erreur « de position », il convient d'ajouter un correcteur intégral.

F. Prieur novembre 2021

Asservissement numérique proportionnel et intégral (PI) :

Il s'agit ici de corriger l'erreur en y adjoignant un terme « intégral » afin de compenser l'erreur de position. La correction sera ainsi d'autant plus grande que la somme des erreurs sera plus grande, lorsque cette erreur sera nulle la valeur de commande sera celle juste nécessaire au maintien de l'erreur nulle (plus d'intégration puisque plus d'erreur…). Notre asservissement étant numérique, nous calculerons le terme intégral grâce à l'échantillon courant u_i et l'échantillon précédent ($u_i.z^{-1}$). On obtient ainsi le schéma de régulation suivant :



Avec :

$$Ki = \frac{Te}{Ti}$$

$$u_{p(n)} = Kp.\varepsilon_{(n)}$$

$$u_{i(n)} = u_{i(n-1)} + Ki.u_{p(n)} = u_{i(n-1)} + Ki.Kp.\varepsilon_{(n)}$$

Remarque : nous ne chercherons pas à modéliser le robot ni à définir sa fonction de transfert. Le réglage du correcteur se fera de façon empirique :

- Régler le temps intégral à sa valeur maximale
  - Augmenter le gain proportionnel Kp jusqu'à la limite de stabilité
  - Diminuer le temps intégral Ti jusqu'à la limite de stabilité
- Pour amortir la réponse :
  - Diminuer Kp
  - Augmenter Ti

L'élève réalisera qu'augmenter le gain permet certes d'améliorer la rapidité de la réponse …mais aura la fâcheuse tendance de déstabiliser la régulation.

6- Apports pédagogiques :

Cette manipulation permet de présenter la notion de « **boucle de retour** » ou de « **boucle fermée** » (j'observe ma sortie et je corrige d'autant plus que l'écart avec ma consigne est grand). Cette notion est bien sûr fondamentale pour appréhender l'automatique et la robotique.

La maquette proposée (deux mBots filoguidés sur un circuit fermé) permettra de mener plusieurs essais avec des couples de valeurs (Kp et Ki) et une analyse critique des réglages.

7- Remarques concernant la mise en œuvre:

Les deux robots (suiveur et meneur) sont tous les deux programmés en mode « suivi de ligne ». Ils ne pourront donc quitter le circuit même en cas de défaut de réglage du correcteur.

Une vitesse moyenne modérée est recommandée : le suiveur doit toujours pouvoir rattraper le meneur. Cette « vitesse moyenne » est d'une certaine façon le « point de repos » de notre asservissement.

Aucun soucis particulier dans les lignes droites. Vous observerez une difficulté dans les virages ou le robot « suiveur » perd facilement de vue le robot « meneur » (celui-ci sortant du champ). Le « suiveur » a alors tendance à accélérer pour « rattraper » le meneur, le risque d'accrochage est alors grand. Une distance « d » située aux alentours de 10 à 15 cm est un bon compromis, le suiveur n'aura pas le temps de rattraper et percuter le meneur avant la sortie de virage et tout se régulera dans la ligne droite suivante.

7- Exemple de programmes en Scratch avec commande « proportionnelle intégrale »:

Deux programmes sont proposés : l'un pour le « meneur » (« *Robot meneur.zip* »), l'autre pour le (ou les) « suiveurs » (« *Robot suiveur avec correcteur PI.zip* »).

Les variables (Ki, Kp…) sont situées en tête de programme. Les valeurs proposées constituent un réglage

8- Vidéo de démonstration :

9- Ressources et liens utiles :