Projet PID

Vacances de la Toussaint

L'objectif de ce projet est de se familiariser avec la notion de PID, très utile pour optimiser le fonctionnement des mécanismes

Contexte

Vous disposez d'une classe *Robot* simulant le déplacement d'un robot sur une ligne droite. Votre objectif est de lui faire parcourir une distance de 20m le plus optimalement possible. Vous autorisez une tolérance de 5cm. Par « le plus optimalement possible », on entend :

- Minimiser le temps de parcours
- Minimiser l'erreur entre la position du robot et la position désirée

Le robot possède une vitesse maximale de 4.2 $m.s^{-1}$ et peut atteindre cette vitesse en 0,5s. Les frottements entre les roues du robot et le sol sont pris en compte. Chaque itération du simulateur correspond à une durée de 0,02s dans la réalité.

Consignes

Étude de la classe *Robot*

1. Regarder le .h pour savoir l'ensemble des méthodes dont vous disposez.
2. La méthode *move* prend en argument une puissance dans $[-1, 1]$ représentant un pourcentage de la vitesse demandée. Ainsi, 0.5 représente une vitesse demandée de 2,2 $m.s^{-1}$, alors que -0.1 représente une vitesse demandée de -0,42 $m.s^{-1}$ (le robot recule). La méthode met à jour la position du robot en fonction de la vitesse demandée à chaque itération. Il faut donc 50 mises à jour par seconde.
   Afficher la vitesse et la position du robot après 1s lorsque la vitesse demandée est de 4,2$m.s^{-1}$.
3. Faites avancer le robot sur une distance de 20m à vitesse maximale. Lorsque la distance est atteinte, la vitesse demandée est 0$m.s^{-1}$. Afficher la position lorsque le robot ne bouge plus et le temps pris pour le déplacement. Que remarquez-vous ? Est-ce optimal ?

Programmation d'un PID

Un PID est un régulateur permettant de fournir une **réponse** optimale en fonction d'une **consigne** et de **mesures**. L'**erreur** est la différence consigne - mesure. L'**erreur totale** est la somme des erreurs présentes et passées multipliées par l'intervalle entre chaque mesure.

4. Créer une classe PID encapsulant la tolérance, la consigne et la consigne. Ajouter les attributs correspondants à la mesure actuelle, la mesure précédente et la somme des erreurs. On ne doit pas pouvoir accéder à ces attributs. Ajouter une méthode vérifiant si la mesure actuelle est suffisamment proche de la consigne en tenant compte de la tolérance.

Un PID comprend 3 coefficients modulant la réponse en fonction des spécifications suivantes :

- Plus l'erreur est grande, plus la réponse doit être grande (composante P **proportionnelle**)
- Plus l'erreur s'accumule, plus la réponse **à terme** doit être grande (composante I **intégrale**)
- Plus la variation d'erreur est grande, plus il faut changer la réponse pour la **stabiliser** (composante D **dérivée**)

5. Modifier la classe PID pour encapsuler en plus trois coefficients : P, I et D

Soient $e(t)$ l'erreur au temps t, $e(t-0,02)$ l'erreur précédente, e_T l'erreur totale et k_P, k_I, et k_D les coefficients P, I et D. La réponse R renvoyée par le PID est :

$$R = k_P e(t) + k_I e_T + k_D \frac{e(t) - e(t-0,02)}{0,02}$$

6. Utiliser la formule ci-dessus pour ajouter une méthode qui renvoie la réponse en fonction de la mesure. *Aide : ne pas oublier que la formule nécessite l'erreur (qui se calcule à partir de la mesure). Ne pas oublier de mettre à jour l'erreur totale.*

Pour simuler la présence d'un capteur sur le robot, on utilise la position du robot à laquelle on ajoute une variable aléatoire basée sur l'exercice 2.9.

7. Utiliser l'exercice 2.9 pour simuler un capteur de distance. Changer les paramètres de $\{0.0, 1.0\}$ à $\{0.0, 0.02\}$. Afficher trois mesures de la distance. Vous devriez observer du bruit dans les mesures.

Implémentation

8. Reprendre le code de l'étape 3. en modifiant la vitesse demandée. La puissance doit être la réponse fournie par le PID en lui fournissant la mesure actuelle.
9. Changer manuellement les valeurs des coefficients du PID pour en apprendre son comportement, en commençant d'abord par faire varier un coeffcient à la fois (les autres à 0).
10. Optimiser le PID.