Centre d'intérêt 3

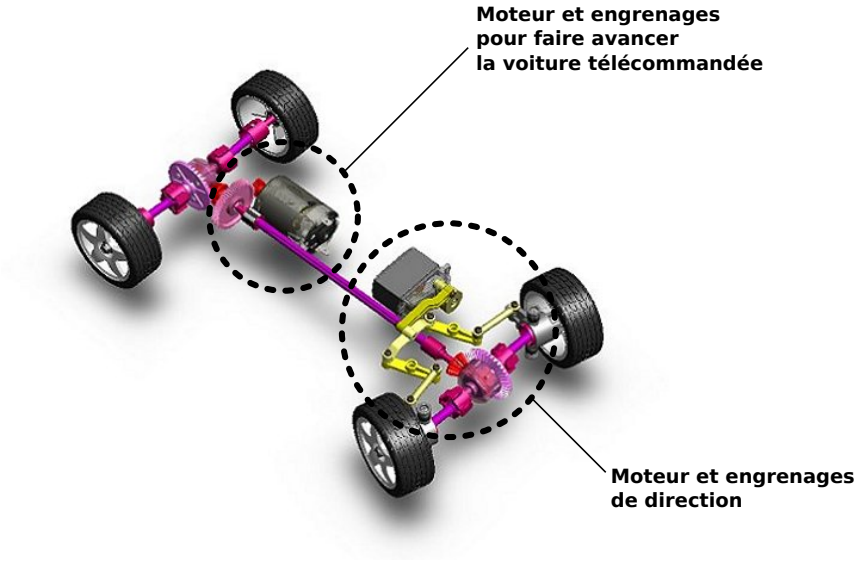
TP : Asservissements

Modélisation numérique d'un système asservi

PSI : Lycée Rabelais

Le système étudié est une voiture télécommandée dont la technologie est présentée ci-dessous. On s'intéressera particulièrement à un asservissement en vitesse permettant de piloter le moteur de propulsion qui permet de faire avancer la voiture.

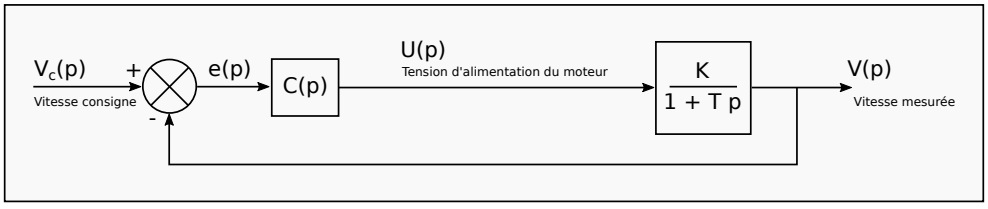
Il est possible de mettre en place un modèle analytique en utilisant notamment la transformée de Laplace mais cette méthodologie est inadaptée lors de non-linéarités. Dans ce cas, une modélisation numérique est alors nécessaire.



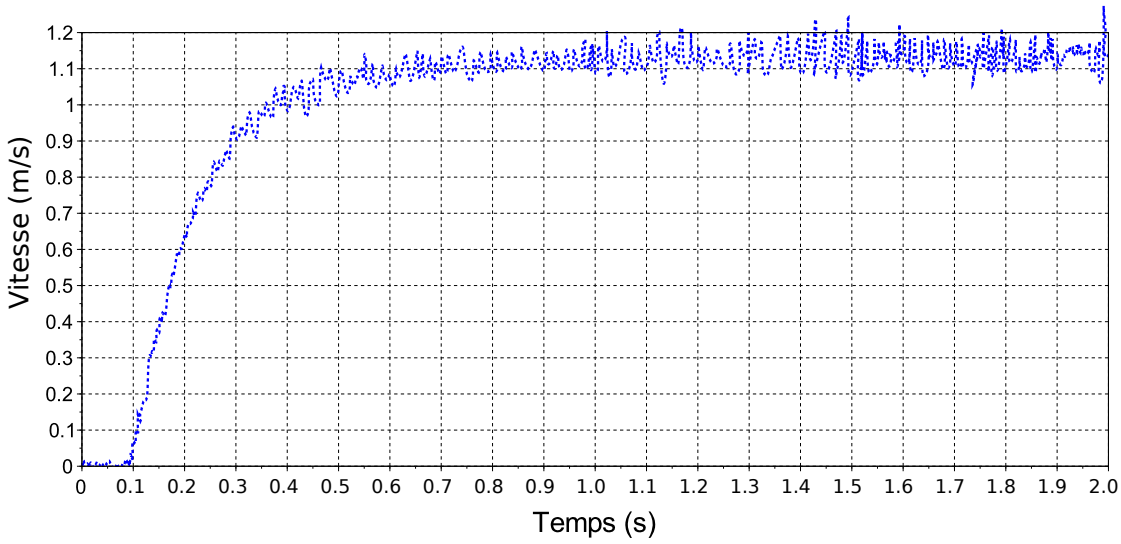
On s'impose le cahier des charges suivant :

Rapidité	Temps de réponse à 5% < 0,15 s
Précision	Erreur statique nulle
Amortissement	Dépassement inférieur à 10%

Le schéma-blocs retenu est le suivant :



1 Identification en boucle ouverte



Question 1. D'une part :

- Je relève $\lim_{t\to+\infty} v(t) = v_\infty \approx 1,15$ m/s.
- Je sais que $v_\infty = K \cdot U_0$.
- Donc $K = \dfrac{v_\infty}{U_0} \approx 0,096$ (m/s)/V.

D'autre part :

- Je relève $t_{r5\%} \approx 0,39$ s.
- Je sais que $t_{r5\%} = 3 \cdot T$.
- Donc $T \approx 0,13$ s.

Question 2. On a :

$$H(p) = \frac{V(p)}{U(p)} = \frac{K}{1 + T \cdot p}$$

Donc $K \cdot U(p) = V(p) + T \cdot p \cdot V(p)$

Et donc $K \cdot u(t) = v(t) + T \cdot \dfrac{\mathrm{d}v}{\mathrm{d}t}(t)$

2 Résolution numérique

On cherche à déterminer l'allure de la sortie en fonction du temps. On suppose dans un premier temps que la vitesse consigne est un échelon d'amplitude $V_0 = 0.8$ m/s.

Pour résoudre les équations différentielles associées à ce système asservi, on propose de discrétiser le problème. On pose alors :

- T_e, le temps d'échantillonnage ou encore appelé pas de temps de la résolution numérique,
- t_i, le i-ème instant discrétisé avec $t_i = i \times T_e$ et donc $t_{i+1} - t_i = T_e$,
- u_i, la valeur approchée de $u(t_i)$. Cela revient à écrire $u_i \approx u(t_i)$.
- $v_i \approx v(t_i)$, , la valeur de la vitesse à l'instant t_i et ainsi de suite pour les autres variables.

Question 3. On peut écrire $\dfrac{\mathrm{d}v}{\mathrm{d}t}(t_i) \approx \dfrac{v_{i+1} - v_i}{T_e}$ et donc $K \cdot u_i = v_i + T \cdot \dfrac{v_{i+1} - v_i}{T_e}$ ce qui donne bien :

$$v_{i+1} = \frac{T - T_e}{T} \cdot v_i + \frac{T_e}{T} \cdot K \cdot u_i$$

Question 4.

```
### Système en boucle ouverte (sans correction)
def SYS(ui,vi):
    return ((T-Te)/T)*vi + (K*Te/T)*ui
```

2.1 Correction proportionnelle

On suppose dans un premier temps que le correcteur est un correcteur proportionnel de telle sorte que $C(p) = K_c$.

On prendra dans un premier temps $K_c = 40$ V/(m/s).

Question 5. $e_i = v_{c_i} - v_i$ et $u_i = K_c \cdot e_i$.

Question 6.

```
def correction_proportionnelle(KC):
    Lv = [0]      ## Liste contenant les vitesses "réelles" vi
    Le = [0]      ## Liste contenant les écarts ei
    Lu = [0]      ## Liste contenant les tension ui
    for i in range(0,n-1):
        vi = Lv[i]
        vci = Lvc[i]
        ei = vci - vi      # Calcul de l'écart
        Le.append(ei)

        ui = KC*ei         # Calcul de la tension d'alimentation
        Lu.append(ui)      # Enregistrement de la valeur

        vip1 = SYS(ui,vi)    # Calcul de Vi+1
        Lv.append(vip1)      # Enregistrement de la valeur

    ### Tracés
    ...
```

Question 7. Il faut $T_e \ll T$.

Question 8. Il faut calculer la FTBF, on obtient : $\text{FTBF}(p) = \dfrac{\dfrac{K \cdot K_c}{1 + K \cdot K_c}}{1 + \dfrac{T}{1 + K \cdot K_c} \cdot p}$

On a donc : $t_{r5\%} = 3 \cdot \dfrac{T}{1 + K \cdot K_c}$ et $v_\infty = \dfrac{K \cdot K_c}{1 + K \cdot K_c} \cdot V_0$

On retrouve les mêmes valeurs numériquement et théoriquement.

Question 9. Non, la tension d'alimentation est beaucoup trop grande. Cette tension doit saturer (**phénomène de saturation**) à 12 V. On modifie :

```
def correction_proportionnelle_saturation(KC,sat):
    Lv = [0]      ## Liste contenant les vitesses "réelles" vi
    Le = [0]      ## Liste contenant les écarts ei
    Lu = [0]      ## Liste contenant les tension ui
    for i in range(0,n-1):
        vi = Lv[i]
```

```
        vci = Lvc[i]
        ei = vci - vi      # Calcul de l'écart
        Le.append(ei)

        ui = KC*ei         # Calcul de la tension d'alimentation

        if ui>sat:
            ui = sat

        Lu.append(ui)      # Enregistrement de la valeur

        vip1 = SYS(ui,vi)    # Calcul de Vi+1
        Lv.append(vip1)      # Enregistrement de la valeur

    ### Tracés
    ...
```

Question 10. ...

2.2 Correction proportionnelle et intégrale

Question 11. Voir cours sur les méthodes numériques !

Question 12.

```
def correction_proportionnelle_intégrale(KC,sat,Ki):
    Lv = [0]      ## Liste contenant les vitesses "réelles" vi
    Le = [0]      ## Liste contenant les écarts ei
    Lu = [0]      ## Liste contenant les tension ui
    I = 0 ## intégrale de l'écart
    for i in range(0,n-1):
        vi = Lv[i]
        vci = Lvc[i]
        ei = vci - vi      # Calcul de l'écart
        Le.append(ei)


        I = I + ei*Te     # Calcul de l'intégrale

        ui = KC*ei + Ki*I

        if ui>sat:
            ui = sat

        Lu.append(ui)      # Enregistrement de la valeur

        vip1 = SYS(ui,vi)    # Calcul de Vi+1
        Lv.append(vip1)      # Enregistrement de la valeur

```

```
    ### Tracés
    ...
```

Question 13. ...

2.3 Correction proportionnelle, intégrale et dérivée

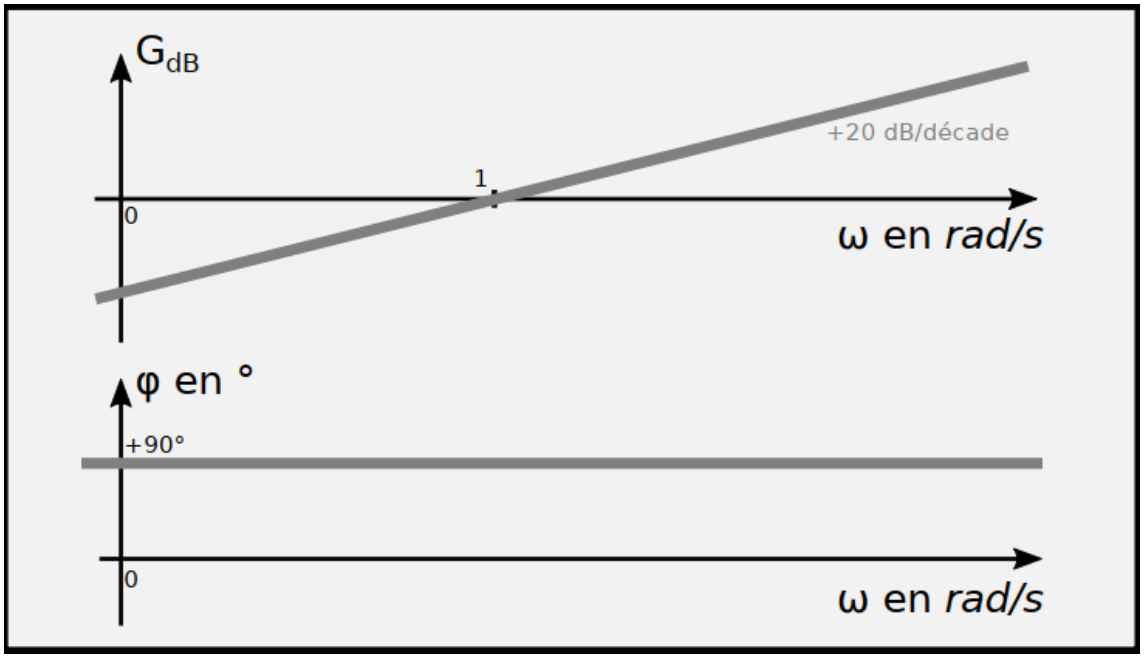
Question 14. Si $i = 0$, il faut alors utiliser le terme e_{-1} qui est inconnu.

Question 15.

```
def correction_proportionnelle_intégrale_dérivée(KC,sat,Ki,Kd):
    Lv = [0]      ## Liste contenant les vitesses "réelles" vi
    Le = [0]      ## Liste contenant les écarts ei
    Lu = [0]      ## Liste contenant les tension ui
    I = 0 ## intégrale de l'écart
    for i in range(0,n-1):
        vi = Lv[i]
        vci = Lvc[i]
        ei = vci - vi      # Calcul de l'écart
        Le.append(ei)


        I = I + ei*Te     # Calcul de l'intégrale
        if i==0:
            Di = 0
        else:
            Di = (Le[-1] - Le[-2])/Te


        ui = KC*ei + Ki*I + Kd*Di

        if ui>sat:
            ui = sat

        Lu.append(ui)      # Enregistrement de la valeur

        vip1 = SYS(ui,vi)    # Calcul de Vi+1
        Lv.append(vip1)      # Enregistrement de la valeur

    ### Tracés
    ...
```

Question 16. ...

Question 17. Tracer le diagramme de Bode associé à l'action dérivée seule : $C_d(p) = K_d \cdot p$.



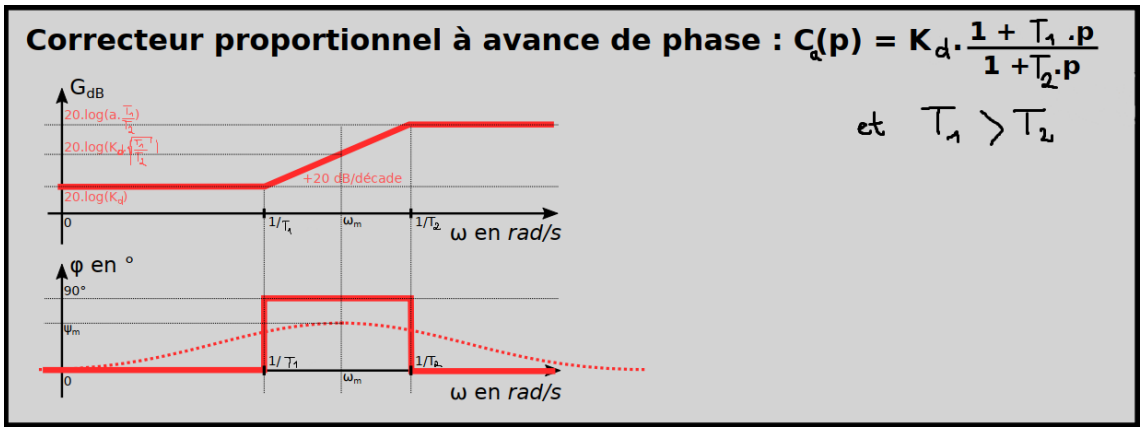
Question 18. En $1,775 - 1,475 = 0,3$ s, je mesure environ 28 "périodes" et ce qui correspond à une fréquence de 93 Hz et donc une pulsation de $\omega_{\text{bruit}} \approx 590$ rad/s.

Question 19. Ici $\omega_{\text{bruit}} \gg 1$ rad/s et donc $G_{dB} \to +\infty$. Le bruit sera donc énormément amplifié.

En pratique et pour éviter ce phénomène d'amplification du bruit, on retient plutôt une correction à avance de phase de la forme suivante :

$$C(p) = K_c + \frac{K_i}{p} + K_d \cdot \frac{1 + T_1 \cdot p}{1 + T_2 \cdot p} \quad \text{où} \quad T_1 > T_2$$

Question 20. Maintenant le gain tend vers une constante quand $\omega_{\text{bruit}} \gg 1$ rad/s.



Question 21. On peut écrire que

$$U(p) = \left(K_c + \frac{K_i}{p} + K_d \cdot \frac{1 + T_1 \cdot p}{1 + T_2 \cdot p}\right) e(p)$$

Ce qui mène à :

$$U(p) + T_2 \cdot p \cdot U(p) = (K_c + K_d + T_2.K_i) \cdot e(p) + (K_c.T_2 + K_d.T_1) \cdot p \cdot e(p) + Ki \cdot \frac{1}{p} \cdot e(p)$$

Ou encore :

$$u(t) + T_2 \cdot \frac{\mathrm{d}u}{\mathrm{d}t}(t) = (K_c + K_d + T_2.K_i) \cdot e(t) + (K_c.T_2 + K_d.T_1) \cdot \frac{\mathrm{d}e}{\mathrm{d}t}(t) + Ki \cdot \int_0^t e(u).du$$

On peut ensuite faire les approximations suivantes :

$$\frac{\mathrm{d}u}{\mathrm{d}t}(t_i) \approx \frac{u_i - u_{i-1}}{T_e}$$

Et

$$\frac{\mathrm{d}e}{\mathrm{d}t}(t_i) \approx \frac{e_i - u_{i-1}}{T_e}$$

Ce qui donne donc :

$$u_i = \frac{1}{1+\frac{T_2}{Te}}\left(\frac{T_2}{T_e}\cdot u_i + (K_c + K_d + T_2.K_i)\cdot e_i + (K_c.T_2 + K_d.T_1)\cdot\frac{e_i - e_{i-1}}{T_e} + Ki\cdot\int_0^{t_i} e(u).du\right)$$

```
def correction_totale(KC,sat,Ki,Kd,T1,T2):
    Lv = [0]      ## Liste contenant les vitesses "réelles" vi
    Le = [0]      ## Liste contenant les écarts ei
    Lu = [0]      ## Liste contenant les tension ui
    I = 0 ## intégrale de l'écart
    for i in range(0,n-1):
        vi = Lv[i]
        vci = Lvc[i]
        ei = vci - vi      # Calcul de l'écart
        Le.append(ei)


        I = I + ei*Te     # Calcul de l'intégrale
        if i==0:
            Dei = 0
        else:
            Dei = (Le[-1] - Le[-2])/Te




        ui = (1/(1 + T2/Te)) * ( (T2/Te)*Lu[-1] + (KC + Kd + T2*Ki)*ei + Ki*I
  + (KC*T2 + Kd*T1)*Dei )

        if ui>sat:
            ui = sat

        Lu.append(ui)      # Enregistrement de la valeur

        vip1 = SYS(ui,vi)    # Calcul de Vi+1
        Lv.append(vip1)      # Enregistrement de la valeur

    ### Tracés
    ...

    ...

```

```
# Appel de la fonction
sat = 12 #V
Ki = 200 #(rad/s)*V/(m/s)
Kd = 2 #V/( (m/s)*(rad/s) )
T1 = 1#s
T2 = 0.01#s
correction_totale(KC,sat,Ki,Kd,T1,T2)
```