Partie 1 - Sciences de l'ingénieur

Robot d'assistance à la mobilité

CORRIGÉ

Sous-partie 1

Question 1.1	Le robot à assistance à la mobilité transforme n'importe quelle chaise en un fauteuil roulant d'intérieur. Il est fin et passe partout. Il est doté de repose-pieds pour être plus confortable et permet de se relever sans trop d'efforts grâce à un vérin électrique.
Question 1.2	Mouvement de 5/1 : **rotation d'axe (O,$\overrightarrow{z_0}$) (de centre O accepté)** TC$\in$ 5/1 : **cercle de centre O et de rayon (CO)** **Voir DR1**
Question 1.3	**Voir DR1**
Question 1.4	Sur le dessin : Longueur [C_0D_0] = **7,7 cm** Longueur [C_1D_1] = **9,1 cm** A l'échelle : Longueur [C_0D_0] = **38,5 cm** Longueur [C_1D_1] = **45,5 cm**
Question 1.5	Course du vérin électrique en phase d'utilisation = **70 mm** Conclusion : **La course est inférieure à 150 mm (voir exigence) donc le vérin est validé.**

Sous-partie 2

Question 1.6

Vmax = 4,3 km/h = 1,19 $m \cdot s^{-1}$

Question 1.7

Voir DR2

Question 1.8

$$r = \frac{\omega_{sortie}}{\omega_{entrée}} = \frac{\omega_{roue}}{\omega_{sortie\ réducteur}} = \frac{Z_2}{Z_1} = \frac{40}{72} = 0{,}55$$

$$r_{global} = \frac{\omega_{roue}}{\omega_{moteur}} = r_{réducteur} \times r = \frac{1}{10{,}215} \times \frac{40}{72} = 0{,}054$$

$$\omega_{roue} = \omega_{moteur} \times r_{global} = \frac{3300 \times 2 \times \pi}{60} \times 0{,}054 = 18{,}79\ m \cdot s^{-1}$$

$$V_{déplacement} = \omega_{roue} \times R_{roue} = 18{,}79 \times \frac{0{,}125}{2} = 1{,}17\ m \cdot s^{-1} = 4{,}23\ km \cdot h^{-1}$$

Question 1.9

A t = 0,8 s :

V_{sim} = 1,12 $m \cdot s^{-1}$

Question 1.10

x = 0,421 - 0,313 = 0,108 m

$$V_{réelle} = \frac{x}{t} = \frac{0{,}108}{(0{,}8-0{,}7)} = 1{,}08\ m \cdot s^{-1}$$

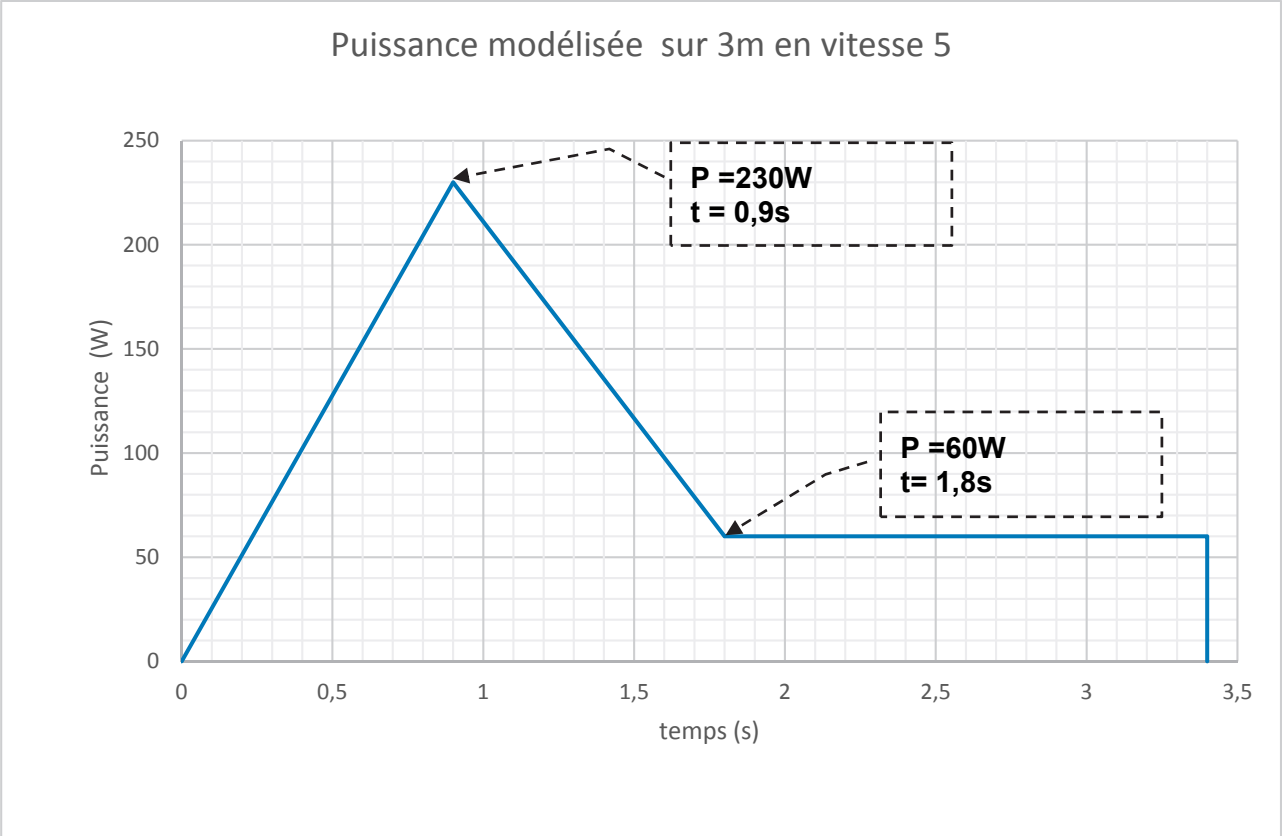
Question 1.11

$$\varepsilon_r = \frac{V_{sim} - V_{réelle}}{V_{réelle}} = \frac{1{,}12-1{,}08}{1{,}08} = 0{,}037 \text{ soit } 3{,}7\%$$

L'écart obtenu est faible ce qui permet de conclure que le modèle multiphysique pourra être utilisé pour les autres vitesses du robot.

Sous-partie 3

Question 1.12



Question 1.13

Distance parcourue par le robot avec la batterie

$$\text{Nb de cycle de 3m} = \frac{227}{0{,}092} = 2467 \text{ cycles}$$

Distance=Nb cycle de 3m ×3=2467 ×3=**7400 m**

Le constructeur annonce 12 km alors que l'on obtient 7,4 km, il n'a pas tenu compte de la phase d'accélération.

Sans prendre en compte la phase d'accélération, la distance parcourue par le robot est de 11,85 km, ce qui est beaucoup plus proche de l'autonomie annoncée par le constructeur.

L'autonomie affichée correspond donc à un cas idéal, sans phase d'accélération, ce qui n'est pas représentatif de l'usage réel du robot.

Question 1.14 La batterie est constituée de 6 éléments en séries

$$UBatterie_{10\%}=6\times3,5=\mathbf{21\ V}$$

Question 1.15 La carte à microcontrôleur est alimentée en 5V, il n'est donc pas possible d'injecter à cette carte une tension supérieure à 5V, la batterie délivre une tension nettement supérieure, d'où la présence du pont diviseur pour réduire et adapter la tension à l'entrée du CAN.

$$UPont_{10\%}=\frac{UBatterie_{10\%}\times6800}{47000+6800}=\frac{21\times6800}{47000+6800}=\mathbf{2,65V}$$

Question 1.16 Valeur du quantum : $q = \frac{V_{FS}}{2^N} = \frac{5}{2^{10}} = 4,88\ mV$

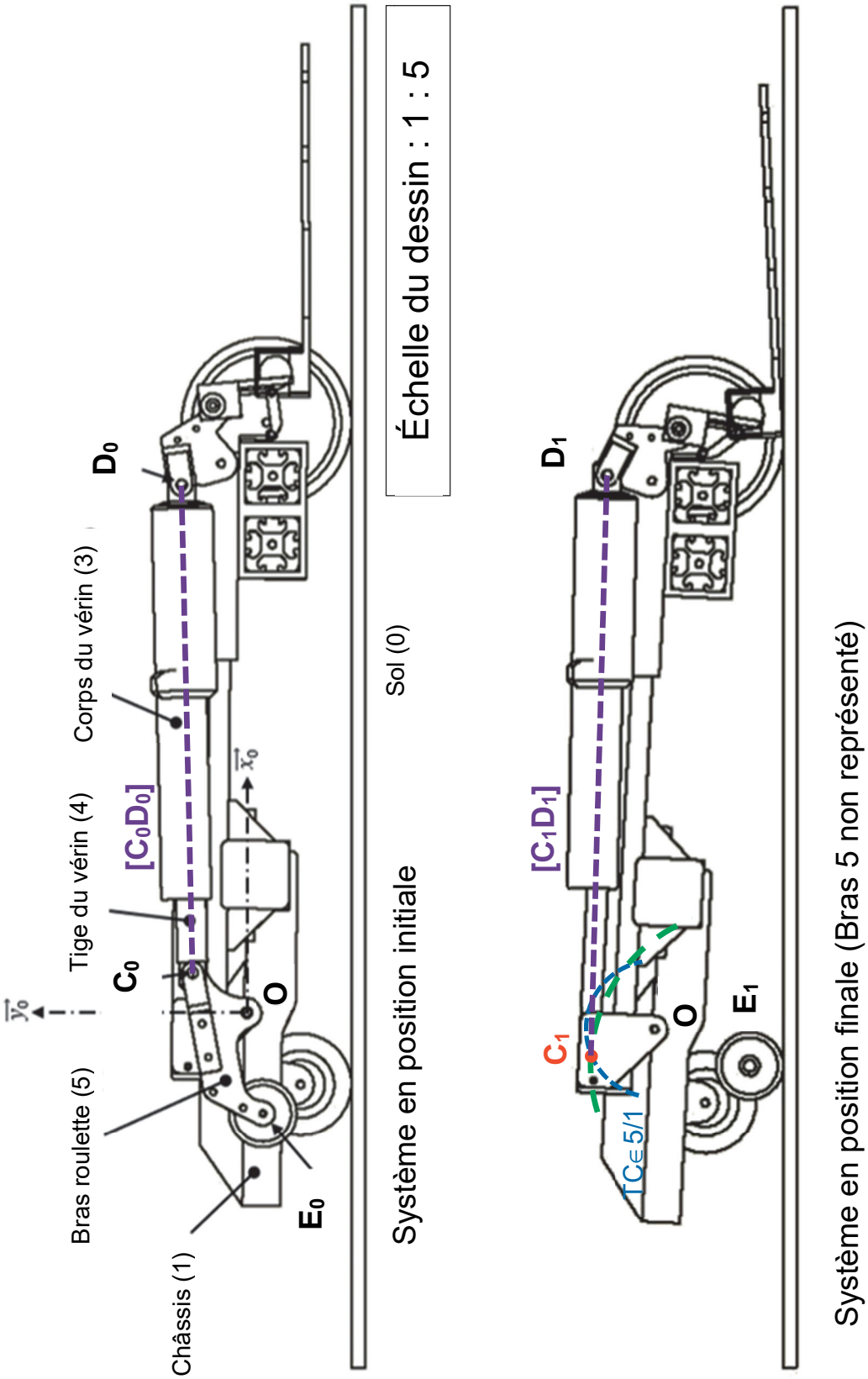
Le nombre $N_{10\%}$ est obtenu quand la batterie délivre une tension de 21V donc $UPont_{10\%}$ vaut 2,65V.

$$N_{10\%} = E\left(\frac{UPont_{10\%}}{q}\right) = E\left(\frac{2,65\times1024}{5}\right) = \mathbf{542}\ (\mathbf{543}\ \text{accepté})$$

Question 1.17 Voir DR3 :

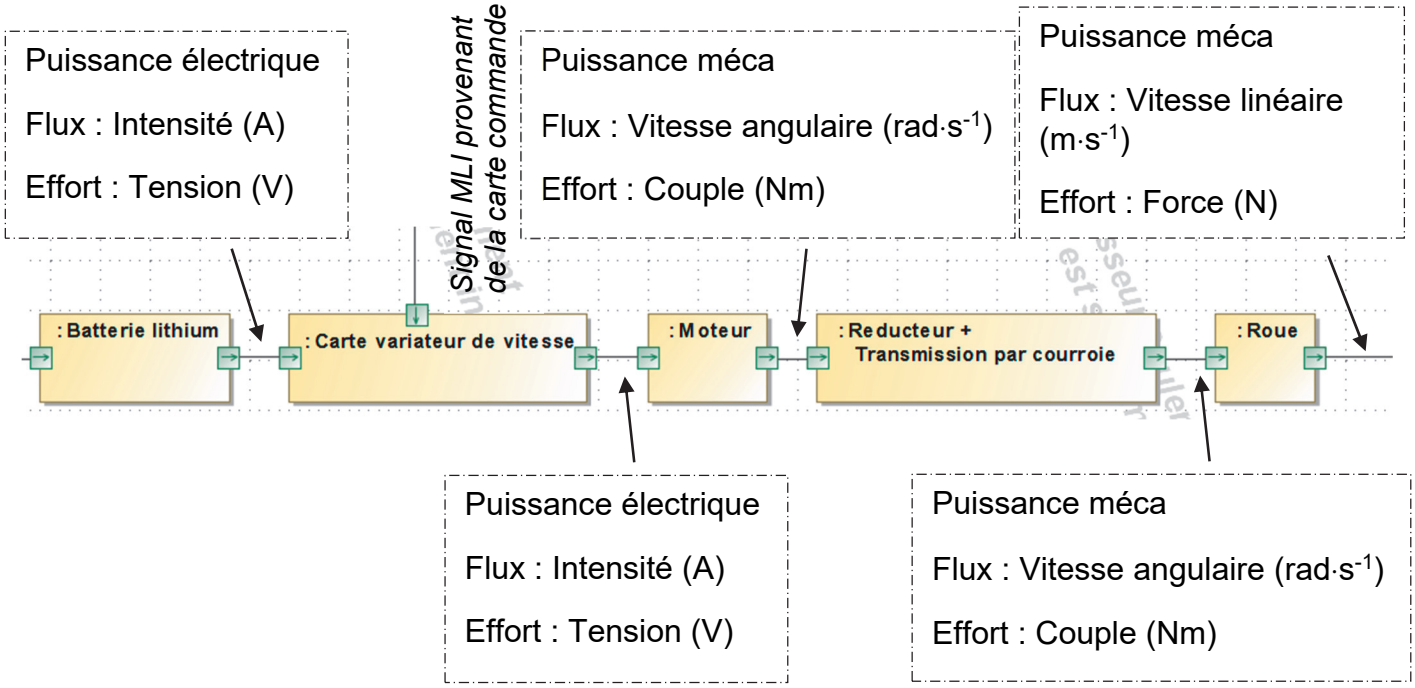
L'autonomie réelle semble moins importante que celle affichée par le constructeur. Il est donc intéressant pour l'utilisateur d'avoir un suivi précis sur l'état de charge de la batterie afin de gérer les recharges indépendamment de la distance parcourue par le robot d'assistance.

Questions 1.2



Document réponse DR2

Question 1.7



Document réponse DR3

Question 1.17

`grovepi.ledBar_setLed(pin,led,state)`	Allume (state=1) ou éteint (state=0) la led spécifiée (1 à 10) du bargraphe connecté à la broche (pin)
`time.sleep(t)`	Permet d'attendre un délai de durée t exprimée en seconde
`SOC_Alerte_10()`	Permet de faire clignoter avec une fréquence de 1Hz la led rouge la plus à gauche du bargraphe

```
 1 import time
 2 import grovepi
 3 # Entrée analogique A2 connectée à UPont
 4 UPont = 2
 5 # Bargraph indicateur du taux de charge batterie connecté à la broche D5
 6 ledbar = 5

 7 grovepi.pinMode(ledbar,"OUTPUT")
 8 # Config. de l'orientation du Bargraph (0 = rouge vers vert, 1 = vert vers rouge)
 9 grovepi.ledBar_init(ledbar, 0)

10 def SOC_Alerte_10():
11 # Clignotement de la Led Rouge à gauche du Bargraphe à une fréquence de 1Hz
12 # ledbar_setLed(pin,led,state),led: which led (1-10), state: off or on (0,1)
13 grovepi.ledBar_setLed(ledbar, 1, 1)
14 time.sleep(.5)
15 grovepi.ledBar_setLed(ledbar, 1, 0)
16 time.sleep(.5)

17 def SOC_Batterie():
18 # Calcul du taux de charge batterie et retourne un entier correspondant au level
19 # à afficher sur le Bargraph
20 # cette fonction est cachée pour plus de lisibilité
21 while 1:
22 # Affichage sur Bargraph de l'énergie dans la batterie
23 N=grovepi.analogRead(UPont)
24 if N > 542:
25     SOC = Calc_SOC_Batterie(N)
26     grovepi.ledBar_setLevel(ledbar, SOC)
27 else :
28 # Appel de la fonction qui permet de faire clignoter la Led Rouge
29 # à gauche du Bargraphe à une fréquence de 1Hz
30     SOC_Alerte_10()
31 # Suite du programme
```