La ligne C du réseau métropolitain Lyonnais et sa crémaillère centrale

Analyse et reproduction en maquette d'une solution unique au monde pour un métro

Louca Malerba - Candidat n°30501

Introduction

A. De la ficelle de la Croix-Rousse au métro C actuel

Jusqu'en 1960 : la "ficelle" de Croix-Rousse :

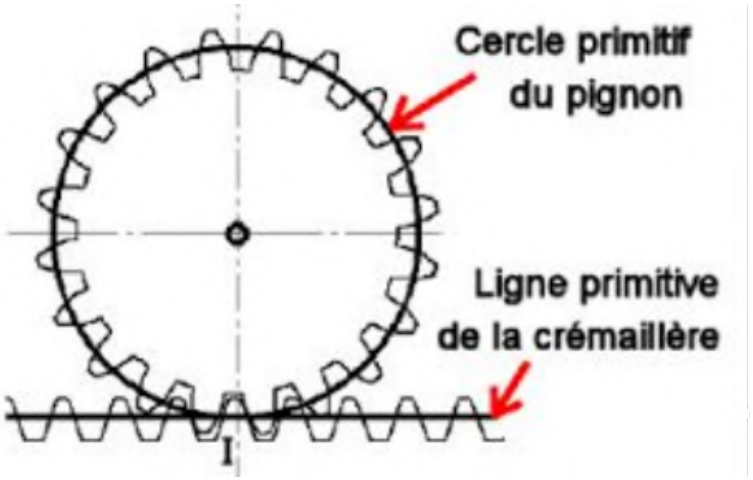
- Besoin de mobilité dès la fin du 19ème siècle.
- Pente maximale de 17,3%

"Ficelle" de la Croix-Rousse - *crédit photo : Albert-Léon Lévy-Lambert - 1908*

Funiculaire similaire présent sur la colline de Fourvière - *crédit photo : Poma.net*

Aujourd'hui : le métro C

Métro C de la Croix-Rousse - *crédit photo : ferro-lyon.net*



Principe de fonctionnement d'une crémaillère (vue de côté) - *crédit image : Académie de Limoge*

B. Généralités sur les trains à crémaillère.

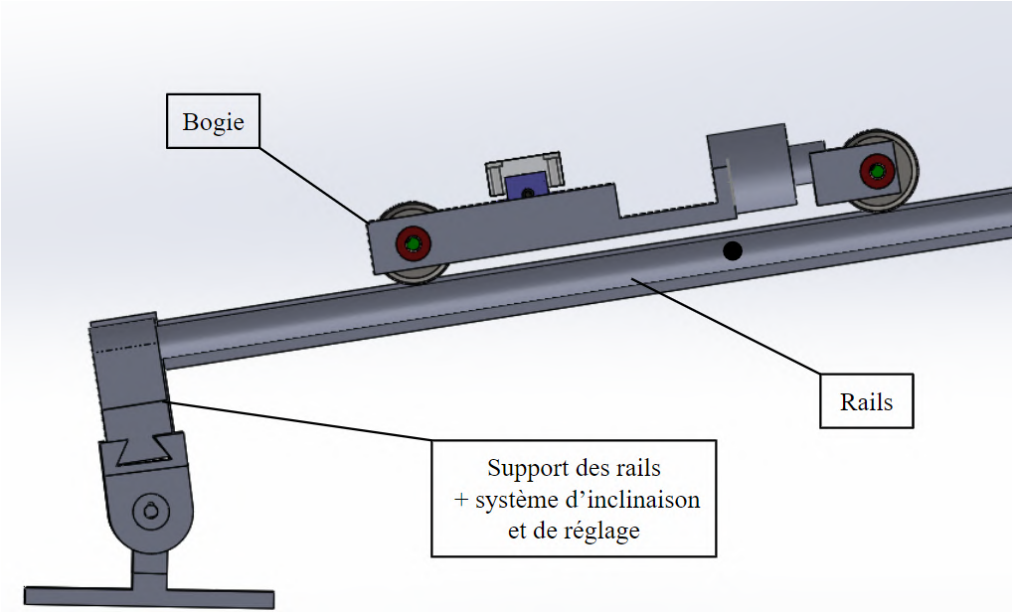
1. Adhérence limitée des roues en acier $\Rightarrow$ pentes praticables en général en dessous de 4%.
2. Chemin de fer à crémaillère = deux rails parallèles en acier + un troisième rail denté, placé entre les rails lisses.
3. Cette solution permet de gravir des pentes allant jusqu'à 48%.

C. Problématique

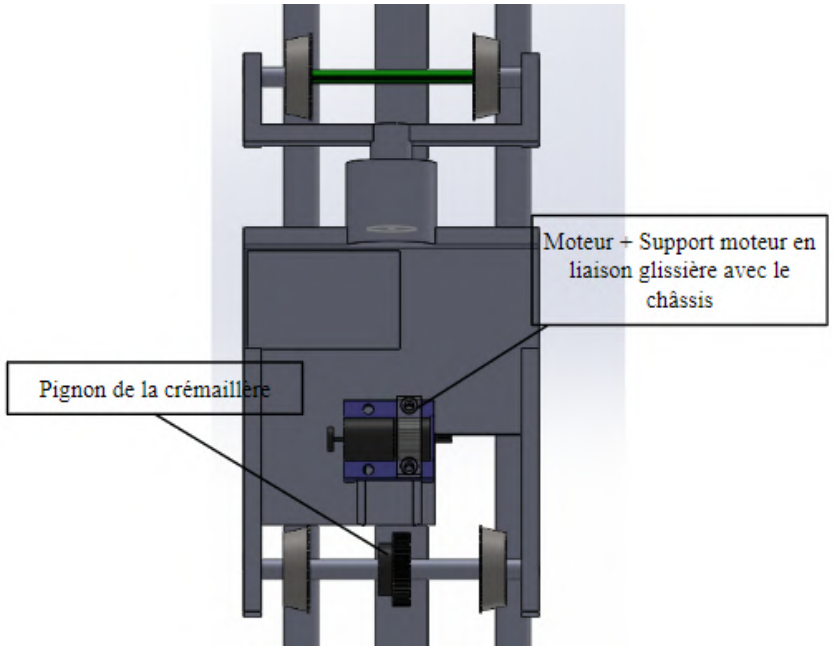
Dans quelles mesures le choix d'un métro à crémaillère a-t-il été un choix pertinent pour l'amélioration de la ligne C du réseau métropolitain ?
Quelle limite de pente peut atteindre par un train à adhérence simple ?

II - Réalisation d'une maquette

Fabrication d'un bogie moteur d'une rame

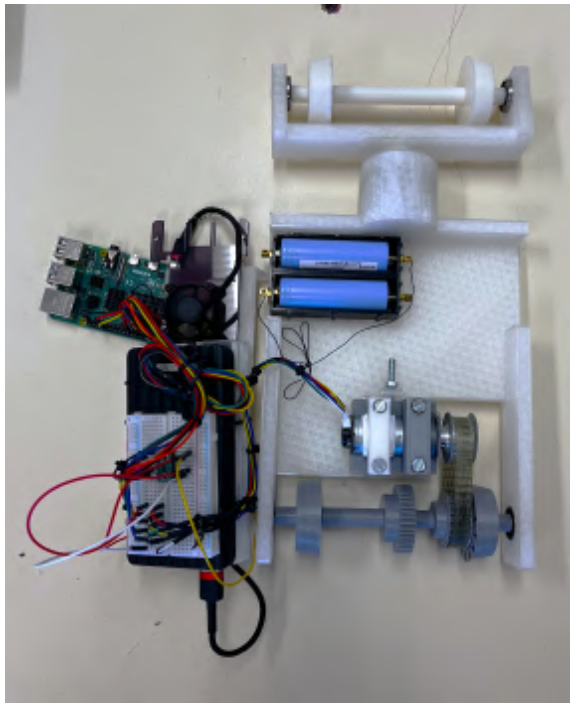


Proposition de modèle 3D, vue de côté

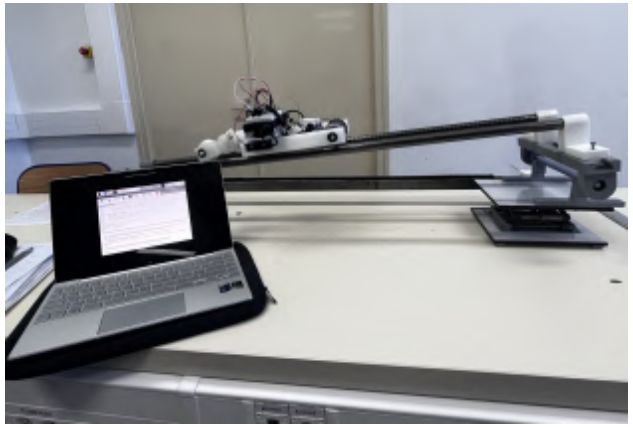


Proposition de modèle 3D, vue de dessus

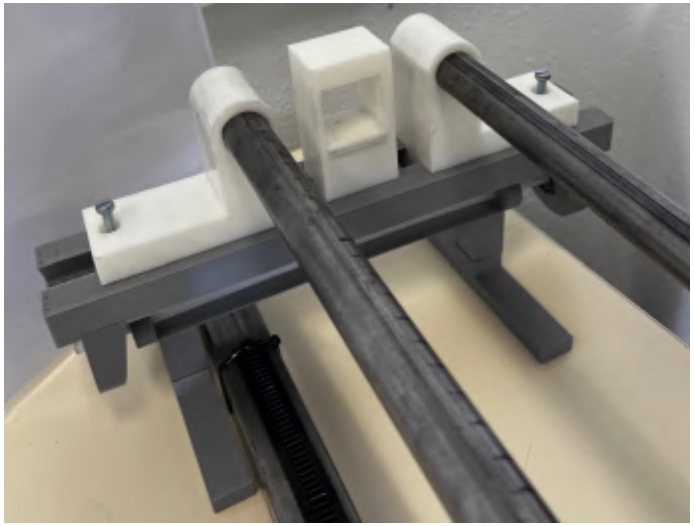
Images de la maquette :



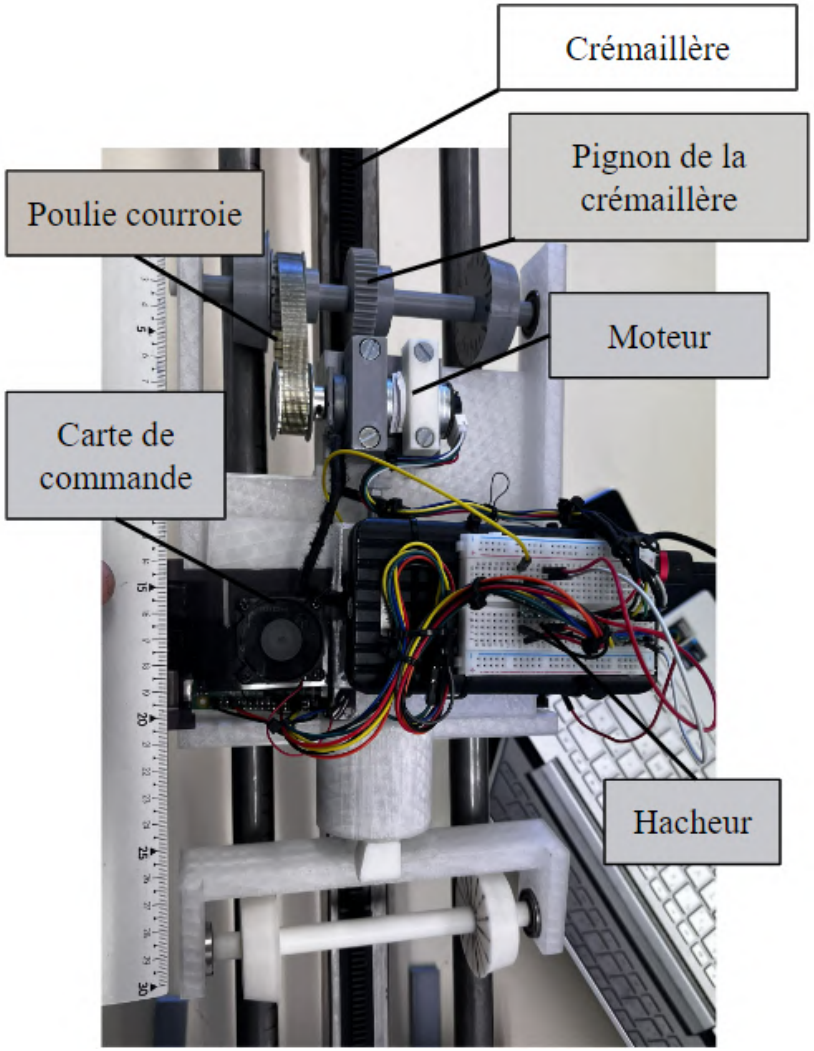
Maquette - vue de dessus



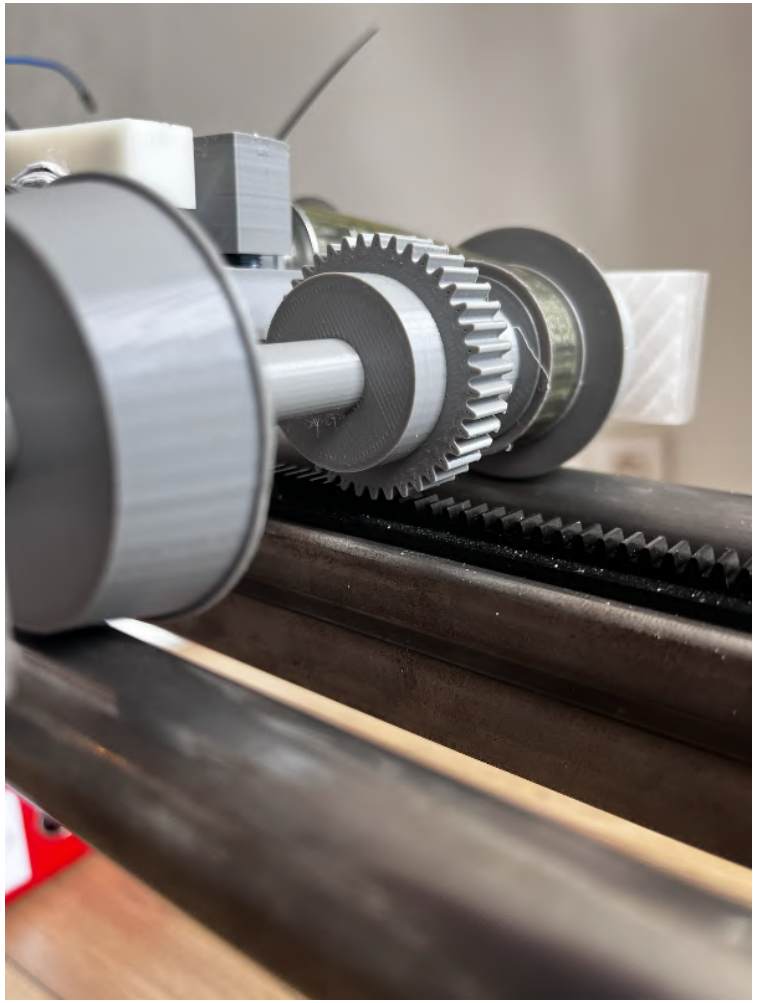
Vue d'ensemble de la maquette



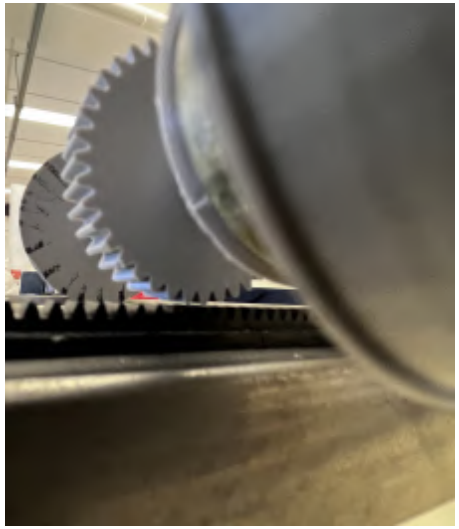
Supports de rail sans crémaillère centrale - (montre le système de réglage et la conception démontable des rails)



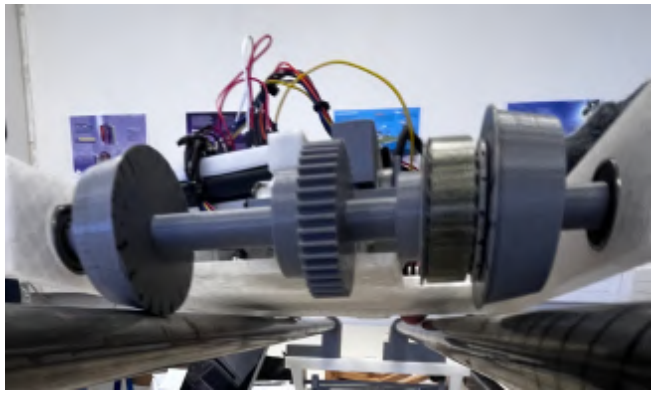
Maquette montée, crémaillère et transmission de la puissance du moteur par système poulie courroie - vue de dessus



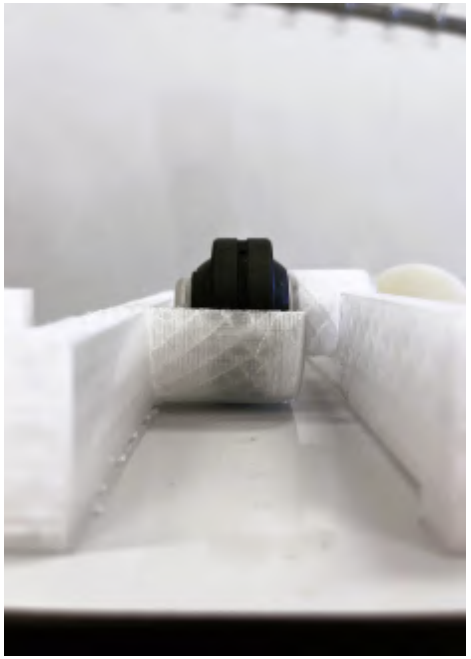
Maquette, crémaillère centrale



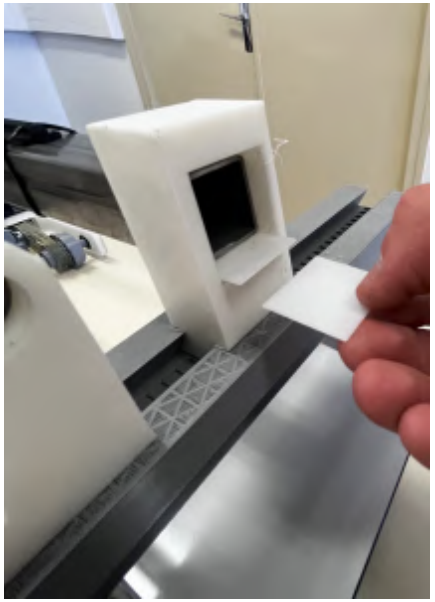
Maquette, mise en évidence de l'hyperstatisme au niveau de la crémaillère



Maquette, mise en évidence de l'hyperstatisme au niveau d'un contact roue/rail



Maquette, rotule comme solution à l'hyperstatisme



Maquette, supports des rails - cales de réglage visibles

III - Expérimentations sur la maquette

Le système doit être capable de gravir une pente élevée.

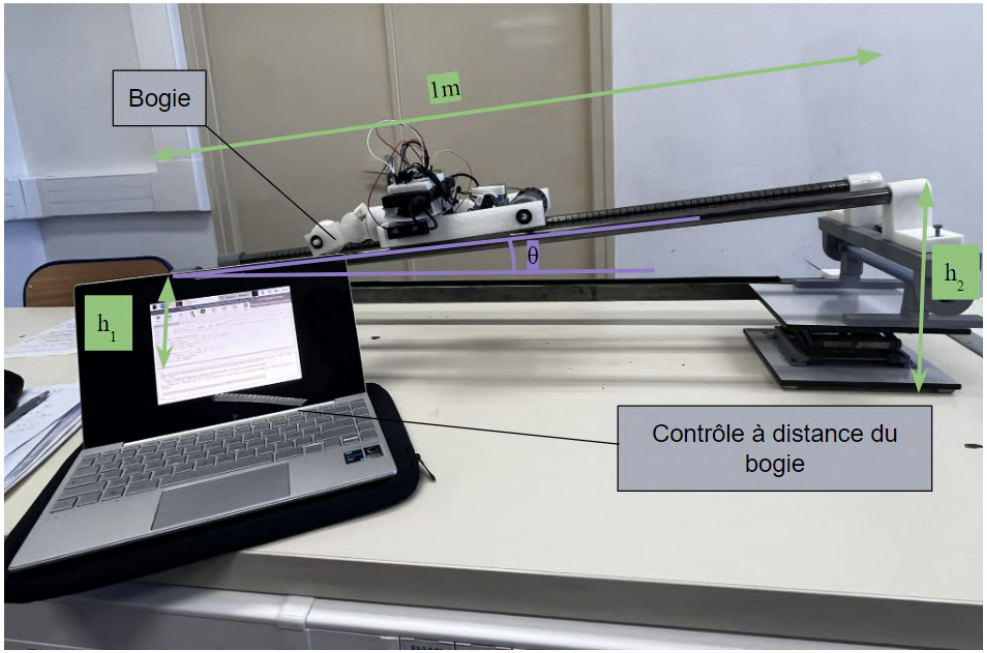
↪ Exigence associée : pente de 17% doit pouvoir être empruntée. (Choix délibéré d'une pente proche du système réel malgré la différence de matériaux)

↪ Objectif de cette partie : valider l'utilité d'une crémaillère et obtention des limites de cette solution.

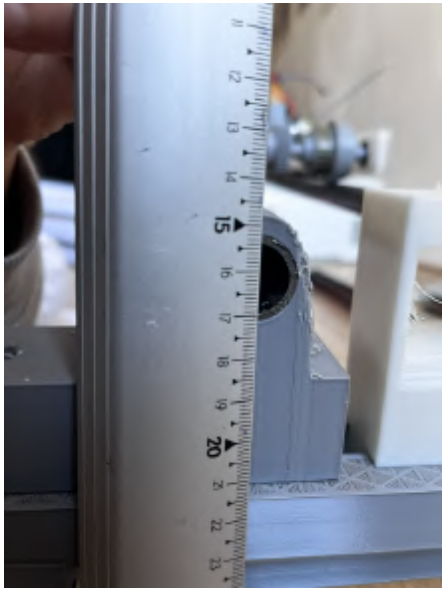
III - A. Coefficient de frottement : Étude statique du système.

Protocole expérimental :

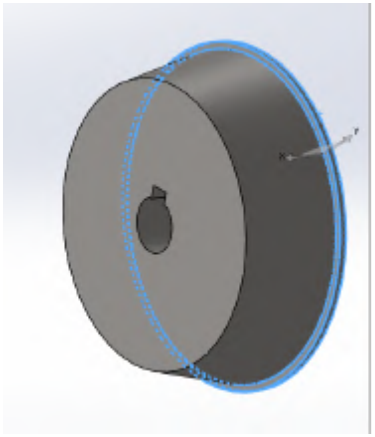
- Enlever la crémaillère centrale.
- Mesurer les hauteurs aux extrémités des rails.
- Poser le bogie dessus et le lâcher le plus délicatement possible. Veiller à ce que les boudins des roues ne soient pas en contact avec les rails (ajouterait un contact ponctuel supplémentaire).
- Répéter l'opération, en changeant la hauteur mm par mm, jusqu'à qu'il soit impossible de garder le bogie fixe sur les rails.



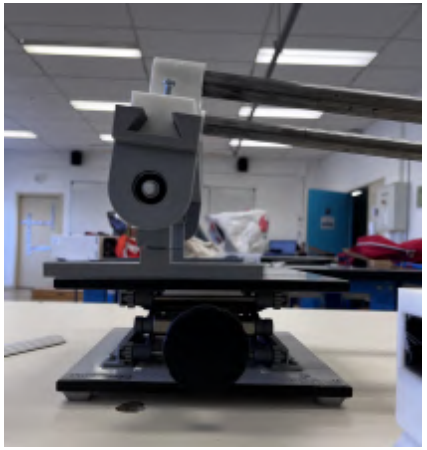
Vue d'ensemble



Lecture d'une valeur de h



Boudin d'une roue



Support élévateur - vue de la molette

Principe fondamental de la statique :

$$[...] \iff f > \frac{sin(\alpha)}{cos(\alpha)} = tan(\alpha) \tag{1}$$

Expérimentalement, sans crémaillère :
- Sur rails secs : $\alpha \approx 12{,}8°$ (22,7%)
- Sur rails mouillés : $\alpha \approx 10{,}3°$ (18,2%)

Donc, sachant qu'on se trouve à la limite de l'angle avant qu'il y ait glissement du bogie :

$$\begin{cases} f = 0.454 \pm 0.002 \; sur\ rails\ secs \\ f = 0.362 \pm 0.002 \; sur\ rails\ mouilles \end{cases} \tag{2}$$

De plus, avec la crémaillère, la pente peut atteindre 26° ($\approx$ 45%)

III - C. Mise en mouvement de la maquette : Étude dynamique.

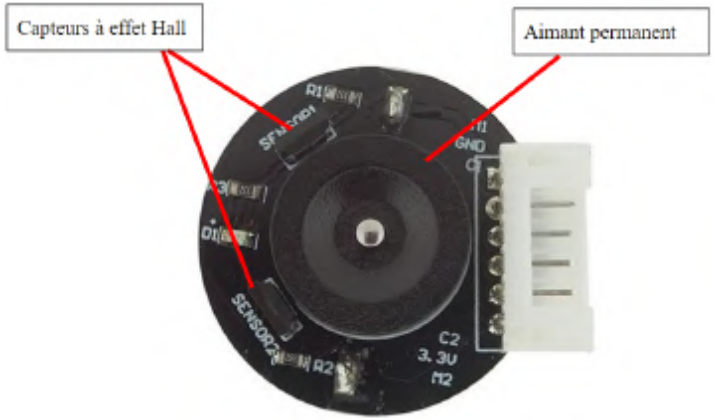
Les pentes obtenues en statique, aussi bien sur pente humide que sur pente sèche respectent le cahier des charges.
Toutefois, ces calculs sont optimistes car ils négligent les effets d'inertie.
L'étude dynamique permet de les prendre en compte.

Il est difficile de modéliser une correspondance entre les performances du système réel et celle de la maquette.
On s'intéresse donc à l'évolution de l'accélération maximale sans glissement en fonction de la pente à gravir sans crémaillère.

Protocole expérimental :

- Poser le bogie sur les rails **sans crémaillère centrale**.
- Placer des repères pour repérer un éventuel glissement plus facilement.
- Lancer un enregistrement vidéo.
- Lancer le programme d'asservissement avec une accélération la plus basse possible.
- Regarder la vidéo, s'il n'y a pas de glissement, recommencer avec une accélération plus grande. Sinon, noter les deux dernières valeurs d'accélération et augmenter la pente puis recommencer.

Extrait d'une vidéo utilisée dans le protocole - (montre les marquages et la vue utilisée)



Capteur à effet Hall - plus de détails en annexe

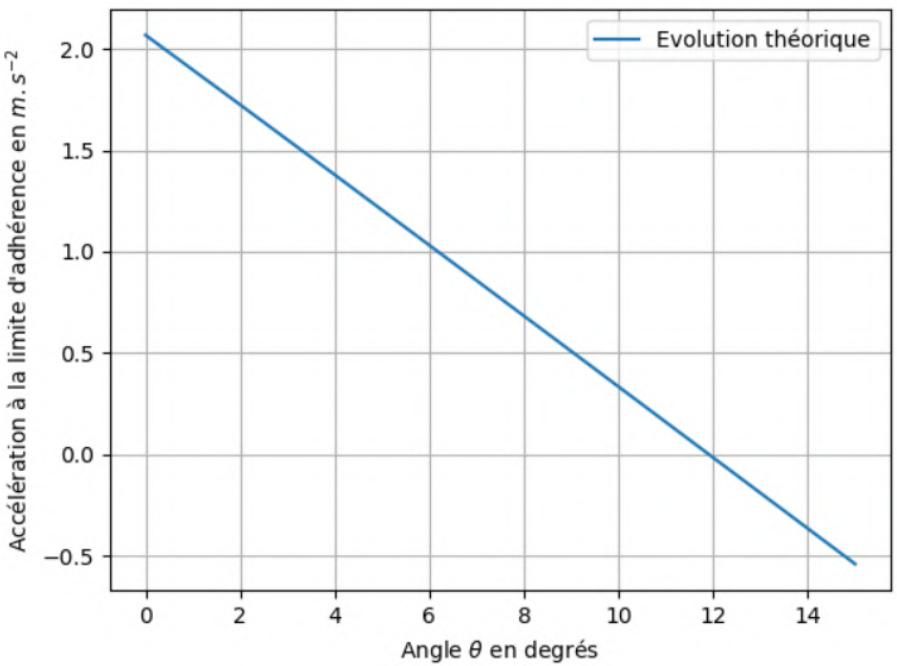
+ Asservissement (en position, vitesse, accélération) grâce à un correcteur PID réglé par la méthode de Ziegler-Nichols

En parallèle, application du principe fondamental de la dynamique

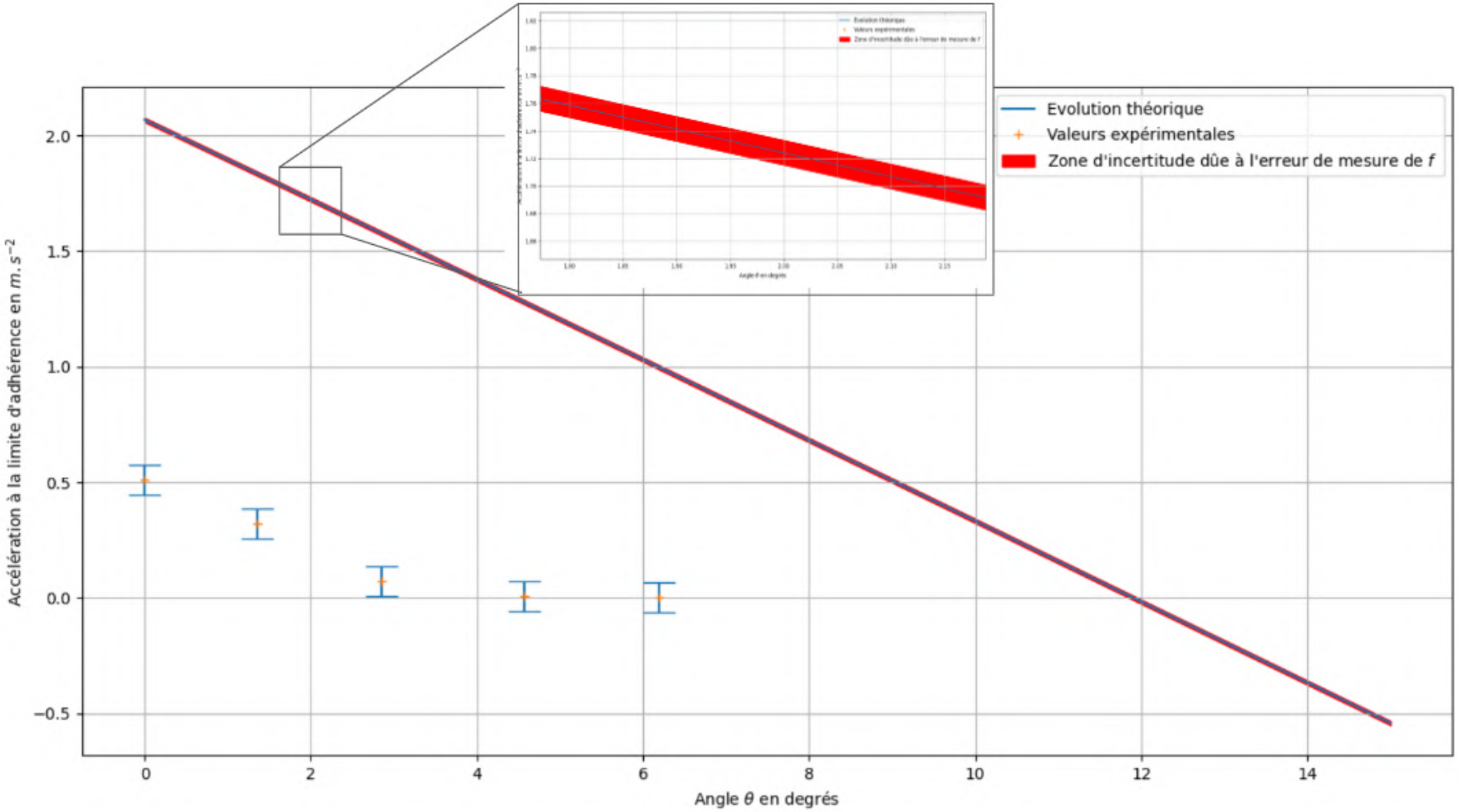
$\Rightarrow$ Valeurs maximales d'accélération pour qu'il y ait roulement sans glissement.

$\Rightarrow$ Condition mathématique :

$$\boxed{\Gamma(G, \Sigma/0) < \frac{b.g.cos(\theta)f}{a+b} - gsin(\theta)}$$



Accélération linéaire maximale en fonction de l'angle d'inclinaison des rails, sans crémaillère centrale



Accélération linéaire maximale en fonction de l'angle d'inclinaison des rails, sans crémaillère centrale

Les résultats de cette expérience ne sont pas satisfaisants.

Causes possibles :

- La liaison pivot arrière non idéale.
- Résolution du capteur de position trop faible. (Fonctionne pour mesurer la position, permet une approximation intéressante de la vitesse, mais incertitude liée à l'accélération trop grande).
- Transmission de puissance imparfaite. (à-coups à faible tension en sortie du hacheur)

Réalisation d'une expérience moins complète :

- Programme modifié pour augmenter la tension très lentement.
- Toujours le même protocole pour vérifier le non-glissement.

$\Rightarrow$ Résultat : angle limite de démarrage sans glissement (montée) :

$$\boxed{\theta_{limite} = 3,3 \pm 0,1^\circ}$$

Source d'erreur possible : Toujours existence d'un à-coup au démarrage.

Même expérience, mais en descente et au freinage :

$\Rightarrow$ Résultat : angle limite de freinage sans glissement en descente :

$$\boxed{\theta_{limite} = 9,5 \pm 0,1^\circ}$$

Source d'erreur possible : La longueur de $1m$ est trop faible pour avoir une décélération très lente.

En conclusion :

Expérimentalement (liaisons non idéales, contrôle de l'accélération imparfait, ...), le système est incapable de gravir une pente de plus de $5,8 \pm 0,2\%$ ($3,3 \pm 0.1°$) sans glissement.

De plus, nécessité d'un coefficient de sécurité + doit être capable de gravir une partie mouillée.

Le système ne répond donc pas aux exigences du cahier des charges.

Il y a donc besoin d'utiliser une crémaillère : l'expérience démontre le fonctionnement du système avec la crémaillère avec des pentes allant jusqu'à 45%.

III - Étude de résistance des matériaux

Le système doit être sécurisé.
La crémaillère et le pignon associé sont très sollicités.
↪ Exigence associée : Résistance du pignon avec un coefficient de sécurité de 3.
↪ Objectif de cette partie : valider le dimensionnement du système réel.

A - Recherche d'informations :

- Peu d'informations disponibles sur le système réel.

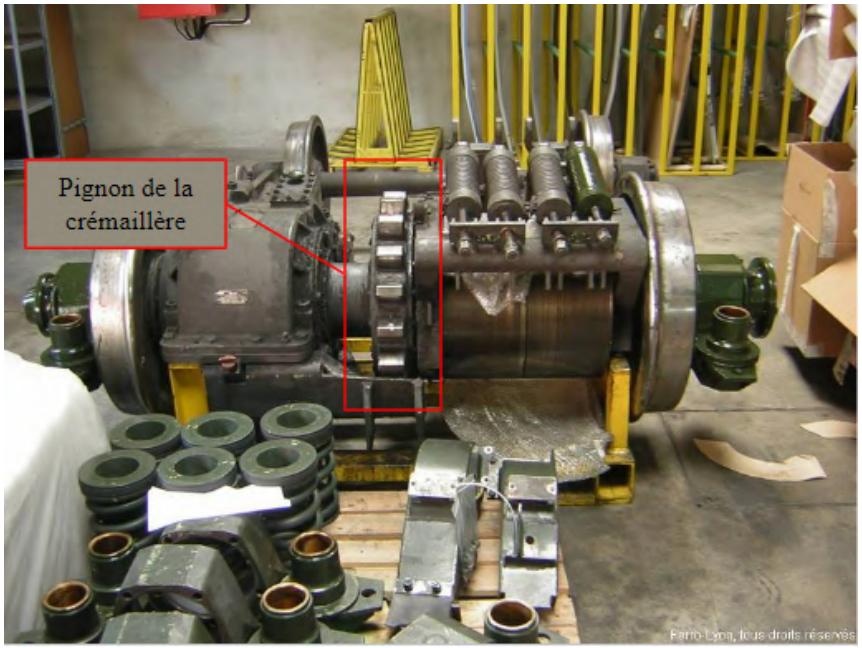


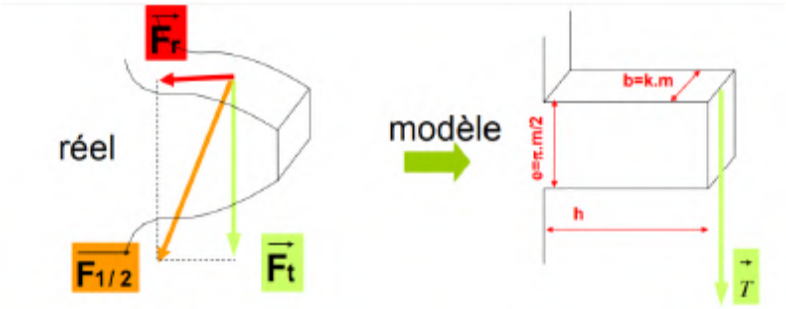
Image du pignon de la crémaillère - *crédit photo : www.ferro-lyon.net - 09/08/2011*

- Diamètre des roues neuves connu : $690mm$
- Photo $\Rightarrow$ 14 ± 1 dents sur le pignon.
- $d = m \times Z \iff m = \frac{d}{Z} = \frac{573}{14} = 40.9mm$
- $\Rightarrow h = m \times 2.25 = 92\text{mm}$

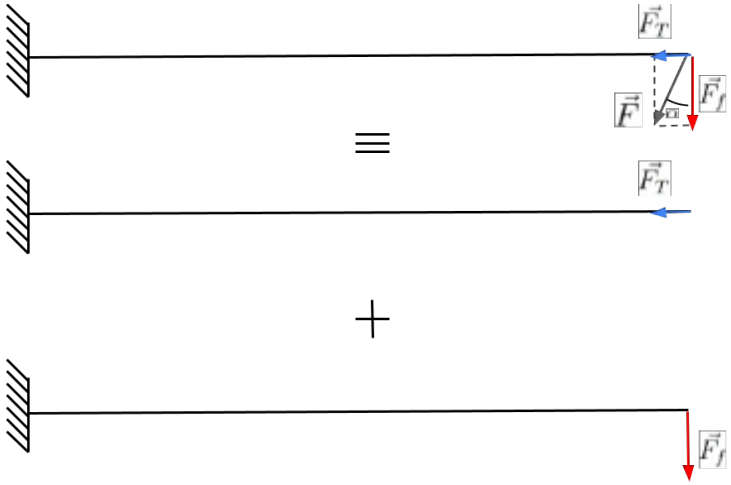
B - Modélisation en poutre droite d'une dent du pignon de la crémaillère.

Hypothèses :

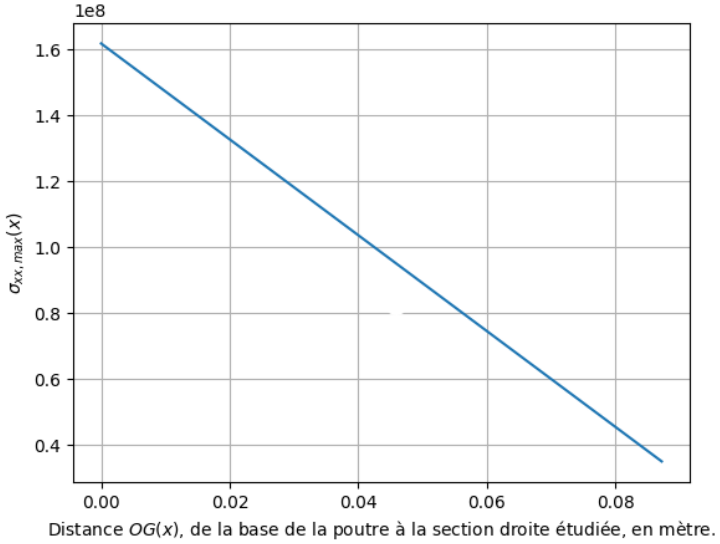
1. On considère une dent comme une poutre de section rectangulaire, "proche" des dimensions d'une dent du système réel.
2. L'effort est orienté de 20° par rapport à l'axe $\vec{y}$.
3. L'ensemble de l'effort de la rame s'applique sur une seule dent.



Modélisation en poutre droite - module du pignon noté m



Principe de superposition



Courbe de $\sigma_{xx,max} = f(x)$

On en déduit la valeur maximale de la contrainte normale :

$$\boxed{\sigma_{xx,max} = 1,62.10^8\ Pa}$$

C - Résolution numérique sous SolidWorks

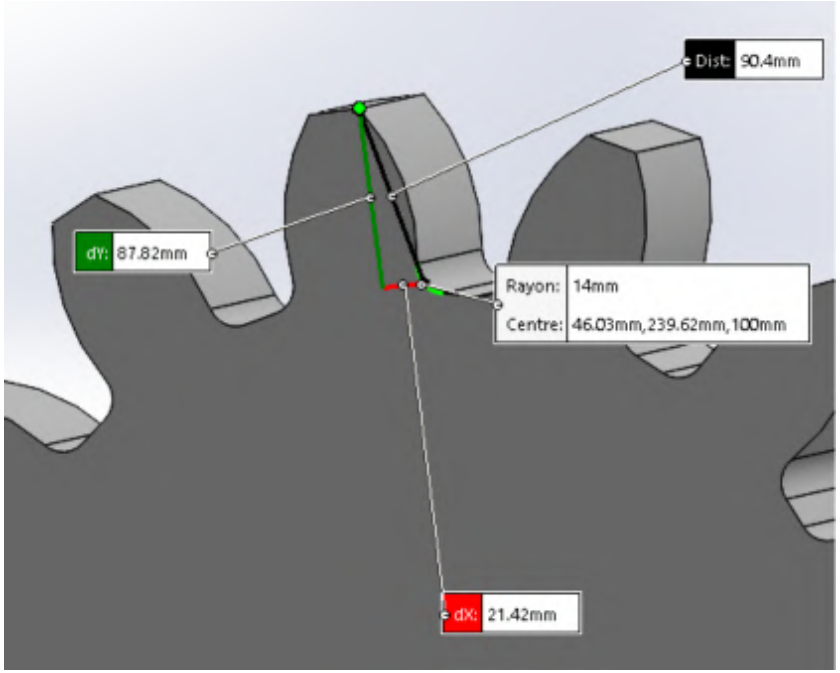


Image du pignon de la crémaillère sous Solidworks

- Matériau appliqué : acier peu allié.
- Effort à 20° comme pour le modèle en poutre droite.
- Conditions les plus défavorables possible :
  - → Effort entier sur une seule dent (irréel mais simplifie les calculs)
  - → Effort appliqué à l'extrémité de la dent.

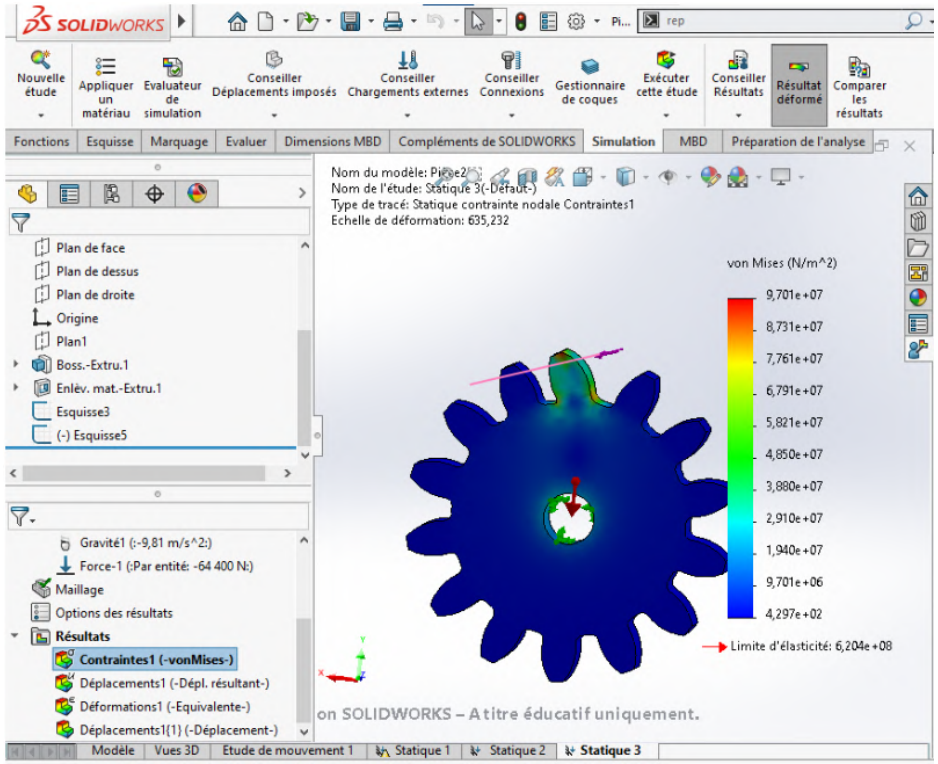
Deux cas considérés :

- Effort maximal réel appliqué sur la crémaillère : $F = 100600\ N$
- Effort pour un freinage entier sur la crémaillère : $F = 64400\ N$

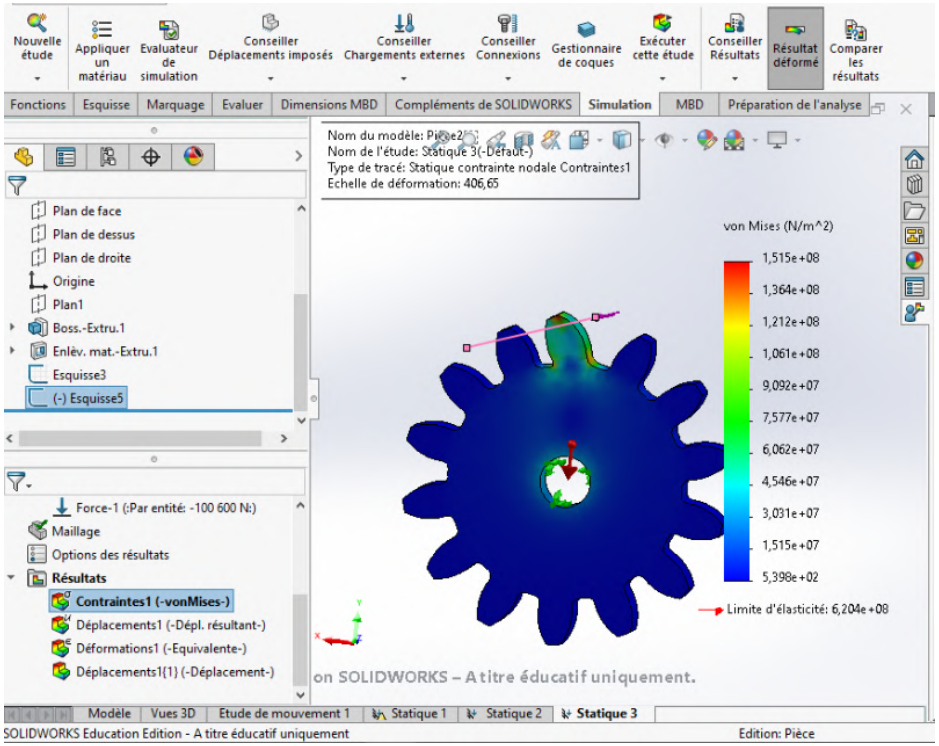
Effort unihoraire à la jante (en adhérence)	2 820 daN
Effort unihoraire à la roue dentée (crémaillère)	6 320 daN
Effort maximal de freinage électrique à la jante (en adhérence)	3 620 daN
Effort maximal de freinage électrique à la roue dentée (crémaillère)	6 440 daN

Tableau récapitulatif des efforts maximaux - *Documentation Alsthom - publié dans la revue Chemins de fer*

On obtient les résultats suivants selon l'exploitation de la rame :



Application des efforts pour l'application de l'effort maximal réel sur la crémaillère - $F = 64400N$



Application des efforts pour l'application de l'effort maximal dans les conditions les plus défavorables (tout le freinage se repose sur une seule dent) - $F = 100600N$

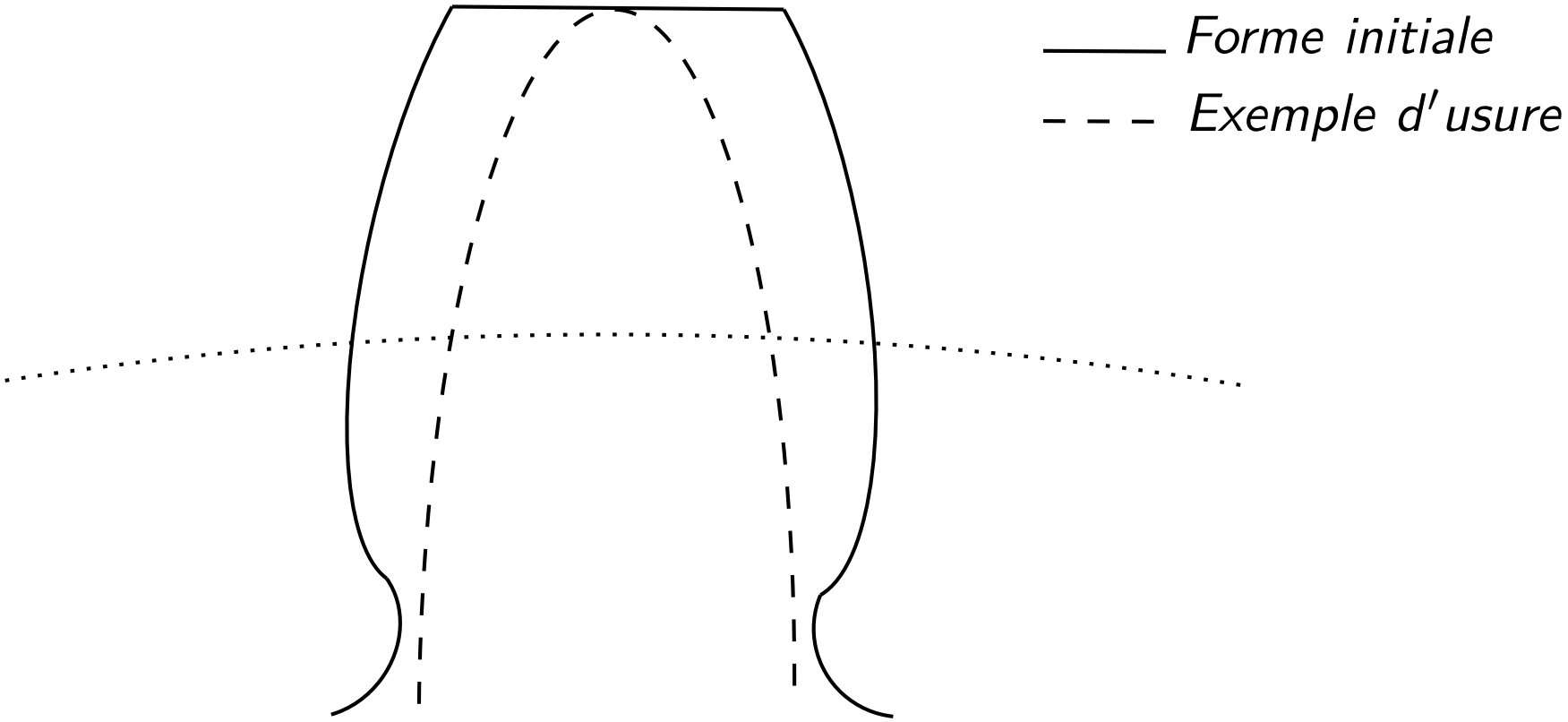
Tableau récapitulatif des résultats :

Méthode	$\sigma_{xx,max}$ (MPa)
Poutre droite, freinage réel	103
Poutre droite, freinage entier sur le pignon	162
SolidWorks, freinage réel	97
SolidWorks, freinage entier sur le pignon	152

Or, SolidWorks nous donne comme donnée pour la limite d'élasticité d'un acier peu allié : $620 MPa$.

$\Rightarrow$ Dans le pire des cas, (très exagéré par rapport à la réalité), un coefficient de sécurité $\boxed{c = \frac{620.10^6}{162.10^6} = 3,8} > 3 \Rightarrow$ Cahier des charges respecté.

À noter : Il est très important d'avoir un coefficient de sécurité élevé sur une rame neuve, car il y a ensuite usure des dents



Annexes

1 Application du principe fondamental de la statique.

BAME: Liaisons sphere-plan:

$$\{\tau^{A}_{sphere-plan\to boggie}\} = \left\{ \begin{array}{c} \overrightarrow{R^{A}_{sphere-plan\to boggie}} \\ \overrightarrow{0} \end{array} \right\}_{A} \tag{1}$$

$$= \left\{ \begin{array}{c} T_{sphere-plan\to boggie}\overrightarrow{x_1} + N_{sphere-plan\to boggie}\overrightarrow{y_1} \\ \overrightarrow{0} \end{array} \right\}_{B} \tag{2}$$

$$\{\tau^{B}_{sphere-plan\to boggie}\} = \left\{ \begin{array}{c} \overrightarrow{R^{B}_{sphere-plan\to boggie}} \\ \overrightarrow{0} \end{array} \right\}_{B} \tag{3}$$

$$= \left\{ \begin{array}{c} T_{sphere-plan\to boggie}\overrightarrow{x_1} + N_{sphere-plan\to boggie}\overrightarrow{y_1} \\ \overrightarrow{0} \end{array} \right\}_{B} \tag{4}$$

$$\{\tau^{C}_{sphere-plan\to boggie}\} = \left\{ \begin{array}{c} \overrightarrow{R^{C}_{sphere-plan\to boggie}} \\ \overrightarrow{0} \end{array} \right\}_{C} \tag{5}$$

$$= \left\{ \begin{array}{c} T_{sphere-plan\to boggie}\overrightarrow{x_1} + N_{sphere-plan\to boggie}\overrightarrow{y_1} \\ \overrightarrow{0} \end{array} \right\}_{B} \tag{6}$$

$$\{\tau^{D}_{sphere-plan\to boggie}\} = \left\{ \begin{array}{c} \overrightarrow{R^{D}_{sphere-plan\to boggie}} \\ \overrightarrow{0} \end{array} \right\}_{D} \tag{7}$$

$$= \left\{ \begin{array}{c} T_{sphere-plan\to boggie}\overrightarrow{x_1} + N_{sphere-plan\to boggie}\overrightarrow{y_1} \\ \overrightarrow{0} \end{array} \right\}_{B} \tag{8}$$

Pesanteur:

$$\{\mathcal{T}^{G}_{pesanteur\to boggie}\} = \left\{ \begin{array}{c} \overrightarrow{R^{G}_{pesanteur\to boggie}} \\ \overrightarrow{0} \end{array} \right\}_{G} \tag{9}$$

Après le changement du points d'application de chaque torseur des actions transmissibles, on obtient la relation suivante grâce au principe fondamental de la statique.

$$\left\{ \begin{array}{l} mgsin(\alpha) + 2T_{sphere-plan\to boggie} = 0 \\ -mgcos(\alpha) + 4N_{sphere-plan\to boggie} = 0 \end{array} \right. \tag{10}$$

$$\Longleftrightarrow \left\{ \begin{array}{l} T_{sphere-plan\to boggie} = \frac{-mgsin(\alpha)}{2} \\ N_{sphere-plan\to boggie} = \frac{mgcos(\alpha)}{4} \end{array} \right. \tag{11}$$

Or, il y a adhérence si et seulement si,

$$|T_{sphere-plan\to boggie}| < f|N_{sphere-plan\to boggie}| \tag{12}$$

$$\Longleftrightarrow \frac{mgsin(\alpha)}{2} < f\frac{mgcos(\alpha)}{4} \tag{13}$$

$$\Longleftrightarrow 2sin(\alpha) < fcos(\alpha) \tag{14}$$

$$\Longleftrightarrow f > 2\frac{sin(\alpha)}{cos(\alpha)} = 2tan(\alpha) \tag{15}$$

Or, expérimentalement:
Sur rails secs: $\alpha \approx 12{,}8°$ (22,7%)
Sur rails mouillés: $\alpha \approx 10{,}3°$ (18,2%)
De plus, avec la crémaillère, la pente peut atteindre 45% (*i.e.*) $\approx 26°$

Donc, sachant qu'on se trouve à la limite de l'angle avant qu'il y ait glissement du boggie:

$$\boxed{\begin{cases} f = 0,454 \; sur \; rails \; secs \\ f = 0,362 \; sur \; rails \; mouillés \end{cases}} \tag{16}$$

Calcul d'incertitude sur le coefficient de frottement :

Calcul de l'incertitude de mesure du coefficient de frottement :

Incertitude sur la mesure des hauteurs extrêmes des rails :

On mesure la hauteur d'un rail avec une règle graduée tous les mm. On considère donc que l'on réalise une mesure dans à $\pm 1mm$ près.

On réalise un calcul d'incertitude de type B:

$u(h_1) = u(h_2) = \frac{\Delta}{\sqrt{3}} \underset{AN}{=} \frac{1}{\sqrt{3}} \approx 0,577mm \approx 0,6mm$

Or , pour obtenir le coefficient de frottement, on s'intéresse à la différence entre ces deux valeurs. Notons $\Delta h = h_2 - h_1 = 227mm$ cette différence.

On applique la loi de propagation des incertitudes pour la somme.

D'où : $u(\Delta h) = \sqrt{(u(h_1))^2 + u(h_2)^2} \approx 0,85mm$

On cherche alors à obtenir le coefficient de frottement : $f = 2 \times tan(\alpha)$.

Or, $tan(\alpha) = \frac{\Delta h}{L}$ où $L = 1m$ est la longueur d'un rail, dont on considère l'incertitude de mesure: $u(L) = \frac{0,5}{\sqrt{3}} = 0,29$.

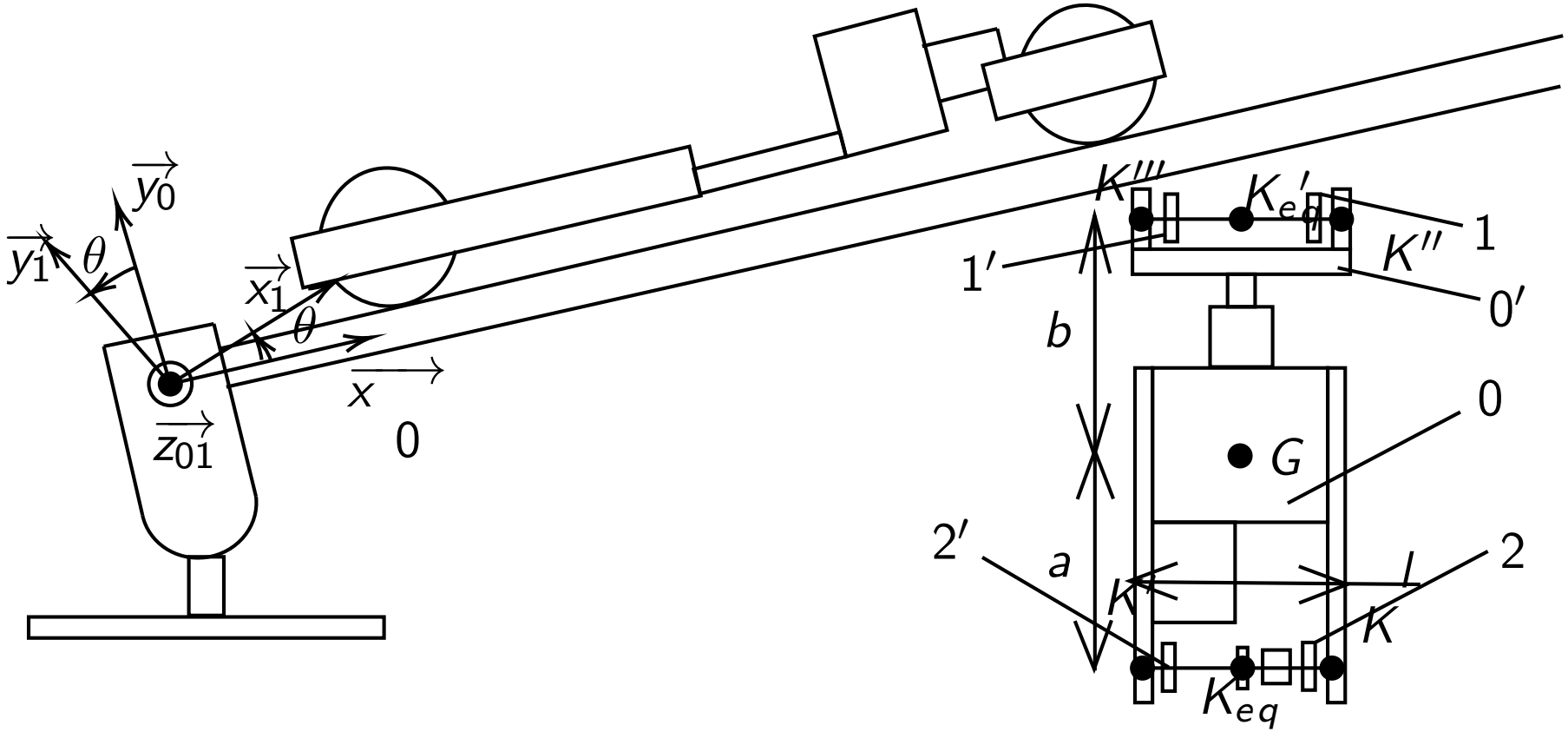
On applique la loi de propagation des incertitudes pour le produit :

Alors :

$u(f) = f \times \sqrt{(\frac{u(\Delta h)}{\Delta h})^2 + (\frac{u(L)}{L})^2} \underset{AN}{=} 0,00169 \approx 0,002$

Donc : $\boxed{f = 0,454 \pm 0,002}$ (sans unité)

Paramétrage Principe fondamental de la dynamique :



Principe fondamental de la Dynamique :

Principe fondamental de la statique appliqué au système sans crémaillère:

Hypothèses:

- On se place dans le cadre du système sans crémaillère: les efforts tangentiels sont en montée, transmis entièrement aux deux roues motrices.
- Les liaisons sont supposées idéales, sans frottement.
- Le problème est supposé plan.

On considère les sous-systèmes:
$\Sigma = \{0, 0'\}; A_1 = \{1, 1', 3'\}; A_2 = \{2, 2', 3\}$ en phase de montée.

Bilan des actions mécaniques extérieures appliqué à Σ:

1. $\{\mathcal{F}_{pesanteur \to \Sigma}\} = \begin{Bmatrix} -m_\Sigma g \vec{y_0} \\ \vec{0} \end{Bmatrix}_G = \begin{Bmatrix} -m_\Sigma g sin(\theta)\vec{x_1} - m_\Sigma g cos(\theta)\vec{y_1} \\ \vec{0} \end{Bmatrix}_G$
2. $\{\mathcal{F}_{A_1 \to \Sigma}\} = \begin{Bmatrix} Y_{A_1}\vec{y_1} \\ \vec{0} \end{Bmatrix}_{K'_{eq}} = \begin{Bmatrix} Y_{A_1}\vec{y_1} \\ Y_{A_1} b \vec{z_1} \end{Bmatrix}_G$
3. $\{\mathcal{F}_{A_2 \to \Sigma}\} = \begin{Bmatrix} X_{A_2}\vec{x_1} + Y_{A_2}\vec{y_1} \\ \vec{0} \end{Bmatrix}_{K_{eq}} = \begin{Bmatrix} X_{A_2}\vec{x_1} + Y_{A_2}\vec{y_1} \\ -a Y_{A_2}\vec{z_1} \end{Bmatrix}_G$

On a de plus le torseur dynamique suivant au point G:

$$\{\mathcal{D}_\Sigma/0\} = \begin{Bmatrix} m_\Sigma \Gamma(G,\Sigma/0)\vec{x_1} \\ \vec{0} \end{Bmatrix}_G$$

Le principe fondamental de la dynamique appliqué à Σ au point G nous donne:

$$\begin{cases} -m_\Sigma g sin(\theta) + X_{A_2} = m_\Sigma \Gamma(G,\Sigma/0) \\ -m_\Sigma g cos(\theta) + Y_{A_1} + Y_{A_2} = 0 \\ Y_{A_1} b - a Y_{A_2} = 0 \end{cases}$$

On résout et on obtient:

$X_{A_2} = m_\Sigma.\Gamma(G,\Sigma/0) + m_\Sigma g sin(\theta)$ l'action tangentielle au niveau <u>des deux roues</u>.

Au niveau d'une roue, on a donc pour action tangentielle:

$T = \frac{X_{A_2}}{2} = \frac{1}{2}(m_\Sigma.\Gamma(G,\Sigma/0) + m_\Sigma g sin(\theta))$

Il y a roulement sans glissement tant que: $T < N * f$ où N est l'action normale sur une roue et où f est le coefficient de frottement déterminé en étude statique.

Le principe fondamental de la statique donne également:

$N = \frac{1}{2}(\frac{m_\Sigma a.b.g cos(\theta)}{a+b})$ où a et b sont des longueurs introduites dans le schéma.

On a donc comme condition de non-glissement:

$$\frac{1}{2}(m_\Sigma.\Gamma(G,\Sigma/0) + m_\Sigma g sin(\theta)) < \frac{1}{2}(\frac{m_\Sigma a.b.g cos(\theta)}{a+b}) \times f$$

$$\iff \Gamma(G,\Sigma/0) < \frac{b.g.cos(\theta) f}{a+b} - g sin(\theta)$$

Calculs de résistance des matériaux :

3 RDM poutre droite

D'après le schéma :

$$\begin{cases} F_f = cos(\alpha)F \\ F_T = sin(\alpha)F \end{cases}$$

Donc : $F_T = sin(\alpha) \times \frac{F_f}{cos(\alpha)} \boxed{= tan(\alpha)F_f}$

Effort n°1 :

On isole la poutre. Bilan des actions mécaniques extérieures:

$$\{\mathcal{F}_{F_T \to \Sigma}\} = \begin{Bmatrix} -F_T\vec{x} \\ \vec{0} \end{Bmatrix}_A$$

Donc :

$$\{\mathcal{F}_{coh}\} = \begin{Bmatrix} -F_T\vec{x} \\ \overrightarrow{G(x)A} \wedge -F_T\vec{x} \end{Bmatrix}_{G(x)} = \begin{Bmatrix} -F_T\vec{x} \\ \vec{0} \end{Bmatrix}_{G(x)}$$

Effort n°2 :

On isole la poutre. Bilan des actions mécaniques extérieures:

$\{\mathcal{F}_{F_f \to \Sigma}\} = \begin{Bmatrix} -F_f\vec{y} \\ \vec{0} \end{Bmatrix}_A$

Donc :

$\{\mathcal{F}_{coh}\} = \begin{Bmatrix} -F_f\vec{x} \\ \overrightarrow{G(x)A} \wedge -F_f\vec{x} \end{Bmatrix}_{G(x)} = \begin{Bmatrix} -F_f\vec{x} \\ (d-x)\vec{x} \wedge -F_f\vec{y} \end{Bmatrix}_{G(x)} = \begin{Bmatrix} -F_f\vec{x} \\ -(d-x)F_f\vec{z} \end{Bmatrix}_{G(x)}$

Donc : Par le théorème de superposition :

$\{\mathcal{F}_{coh}\} = \begin{Bmatrix} -F_T\vec{x} - F_f\vec{y} \\ (x-d)F_f\vec{z} \end{Bmatrix}_{G(x)}$

En prenant :

- $F_f, max = 64400 + 36200$ (cas de l'effort sur la roue dentée + au niveau des roues)
- $y_{max} = \frac{e}{2} = \frac{64,4\times10^{-3}}{2}m$
- $d = 87,2 \times 10^{-3}m$
- $\alpha = 20$
- $I_{G_z} = \frac{100.10^{-3}\times(64,4.10^{-3})^3}{12}$
- $S = 100.10^{-3} \times 64,4.10^{-3}$

On obtient :

- $\boxed{\sigma_{xx,max} = |\frac{-tan(\alpha)F_{f,max}}{S}| + |\frac{-y_{max}(x-d)F_{f,max}}{I_{G_z}}| \approx 1,62.10^8 Pa}$
- $\boxed{\sigma_{xy,max} = |\frac{-F_{f,max}}{S}| = \frac{-64400}{100.64,4.10^{-6}} \approx 1.10^7 Pa << \sigma_{xx,max}}$

  Or, la contrainte équivalente de Von Mises a pour expression :

  $\sigma_{VM,max} = \sqrt{(\sigma_{xx,max})^2 + 3(\sigma_{xy,max})^2}$

  On négligera donc la valeur de $\sigma_{xy,max}$ devant celle de $\sigma_{xx,max}$ nous amenant à réaliser l'approximation: $\sigma_{VM,max} = |\sigma_{xx,max}|$

Méthode de Ziegler-Nichols pour le réglage du PID - *Traduction depuis l'anglais par mes soins de l'article Ziegler-Nichols Tuning Rules écrit par Microstar Laboratories* :

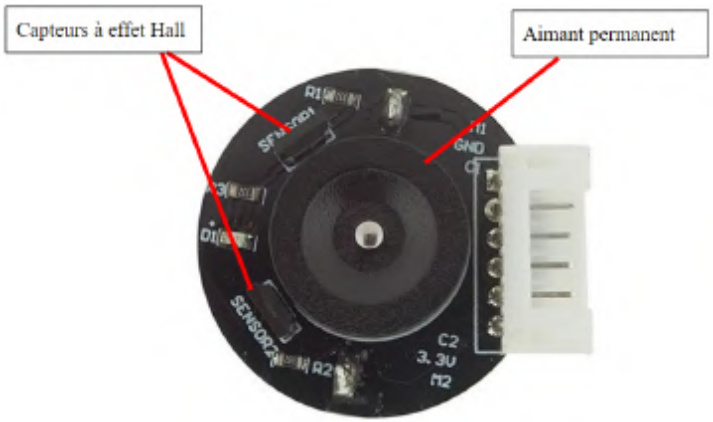
"La méthode de Ziegler-Nichols est une méthode heuristique de réglage d'un correcteur PID. C'est-à-dire qu'il s'agit d'une méthode permettant d'avoir des valeurs "assez bonnes" pour les trois valeurs de gain:
- K_p le gain proportionnel
- T_i la constante d'intégration
- T_d la constante de dérivation"

Il y a plusieurs valeurs possibles pour cette méthode, selon le type de réglage souhaité. Celui que j'ai utilisé est celui référé comme étant "la règle de Ziegler-Nichols classique" dans l'article:

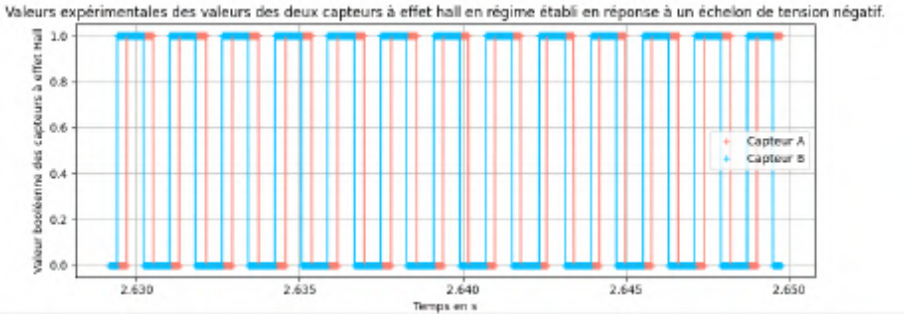
$$K_p = 0.6Ku;\ T_i = 0.5T_u;\ T_d = 0.125T_u.$$

où : K_u et T_u sont respectivement: la première valeur de gain proportionnel pour laquelle il y a oscillation (on a $T_u = T_d = 0$ lors de cet essai) et la période des oscillations.

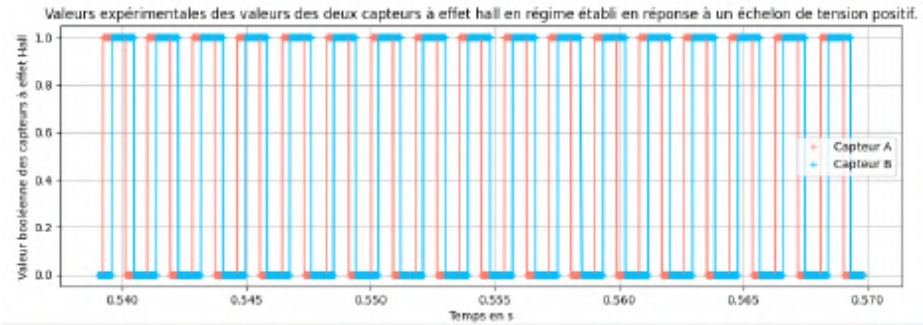
Récupération de la position à partir des capteurs à effet Hall :



Capteurs à effet Hall de la maquette - *Source : gotronic.fr*



Évolution de la sortie du codeur - Sens antihoraire



Évolution de la sortie du codeur - Sens horaire

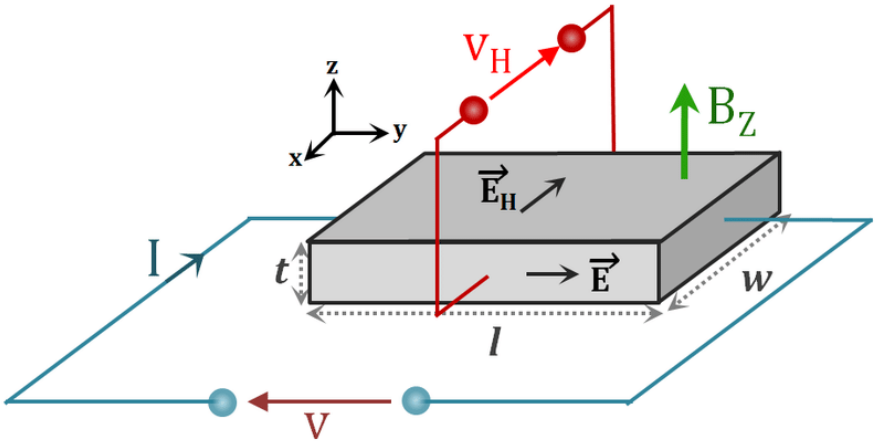


Schéma principe de fonctionnement d'une sonde à effet Hall - *Source : Laure Arbenz via https://www.researchgate.net/*

Image supplémentaire : bogie vue de dessus.

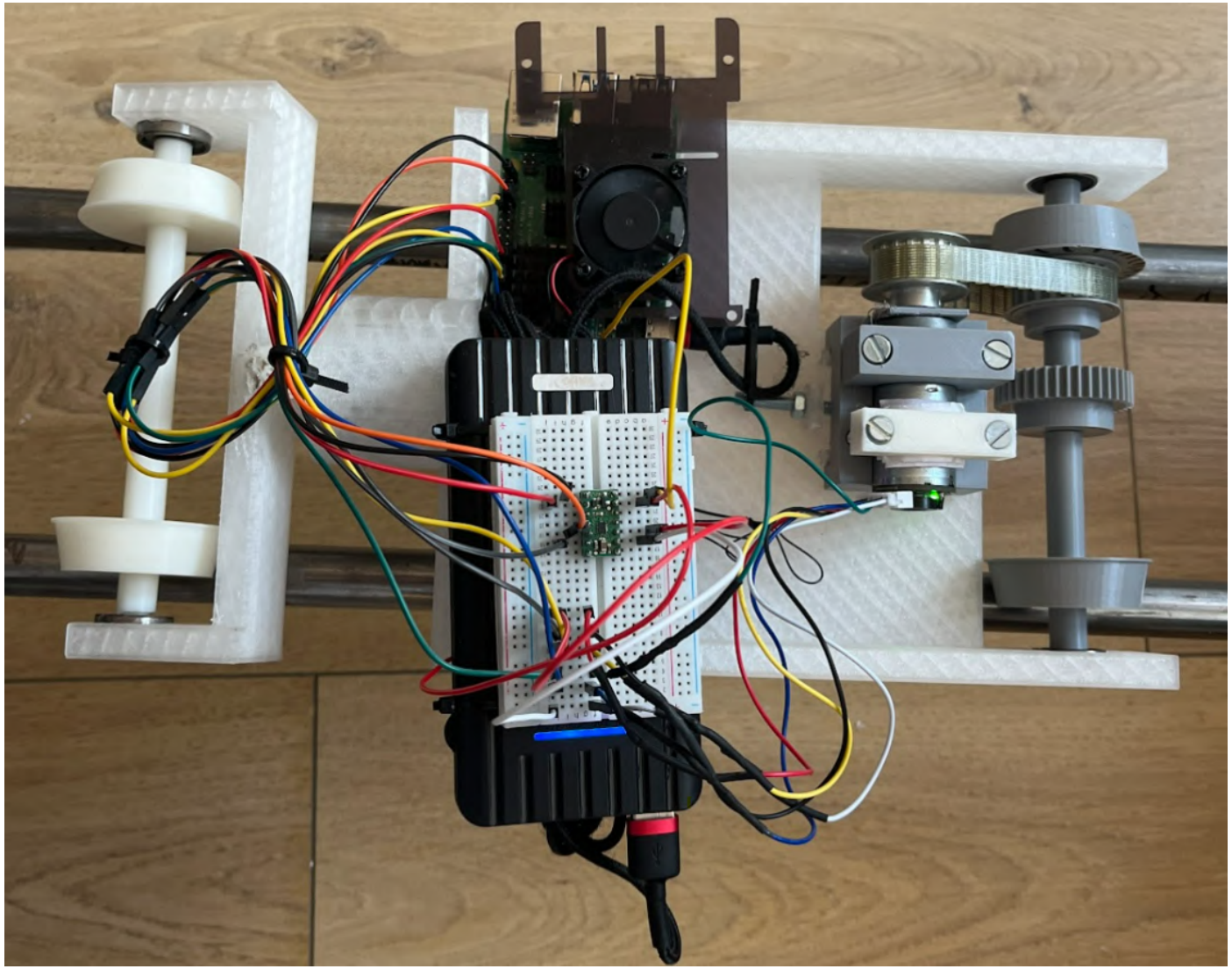


Image supplémentaire : Système de réglage de distance des rails vue de dessus.

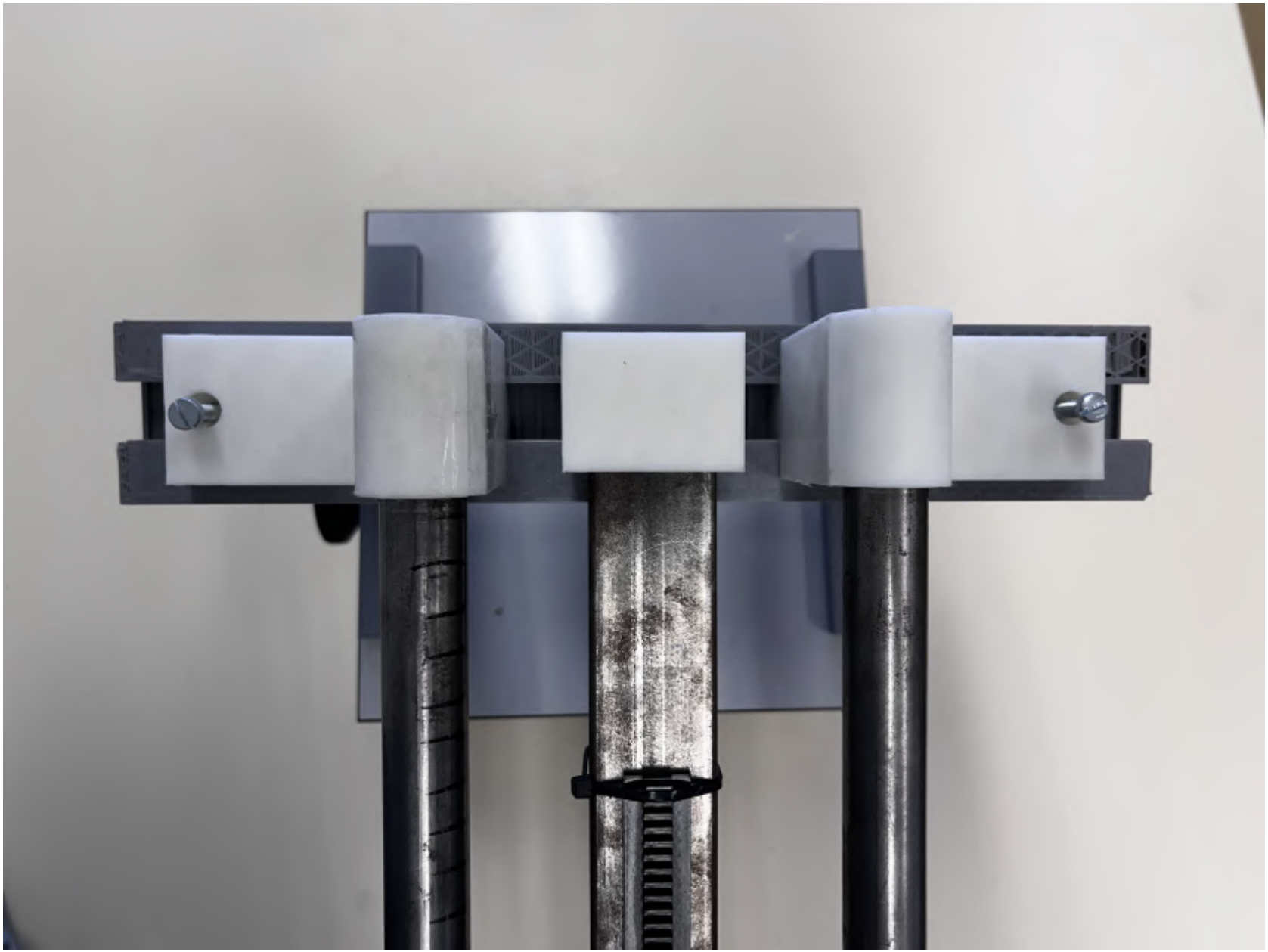


Image supplémentaire : Hyperstatisme, réglage des rails trop proches.

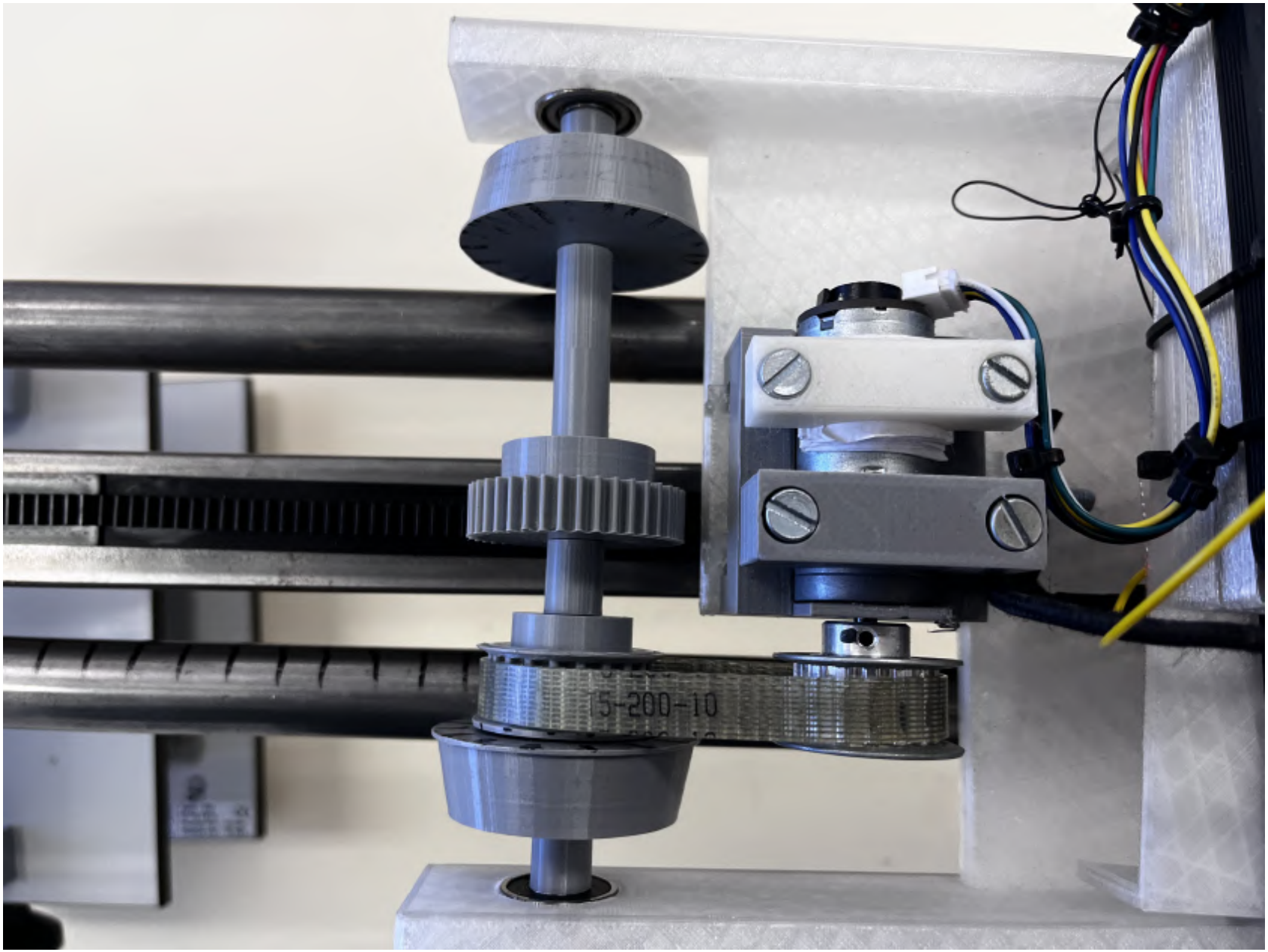


Image supplémentaire : Support élévateur déployé.

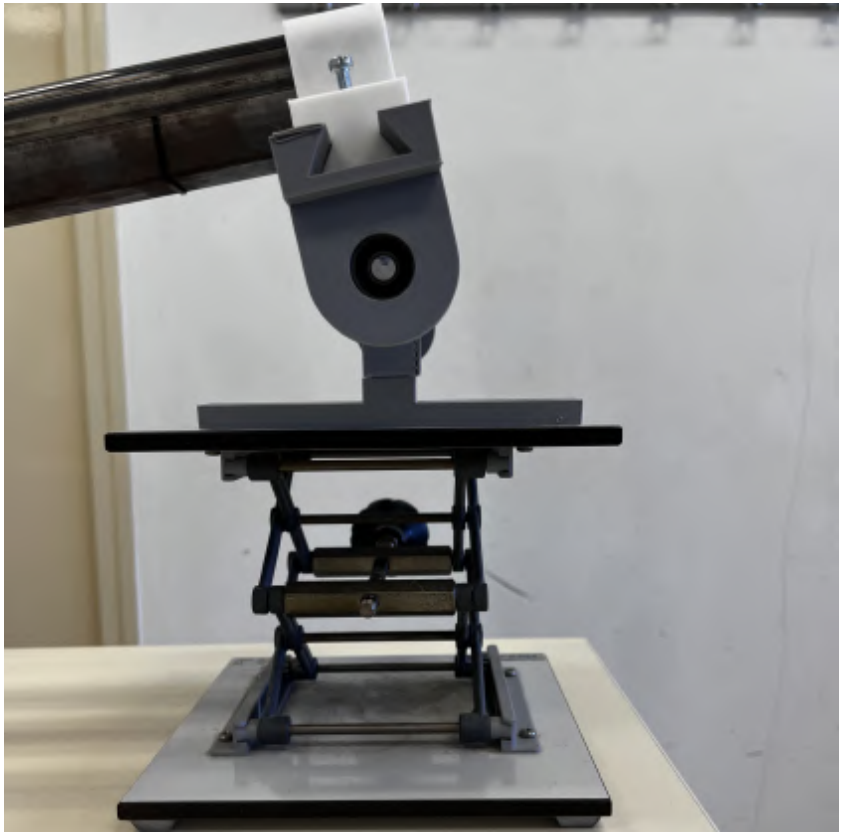


Image supplémentaire : Support de rails vue de derrière.

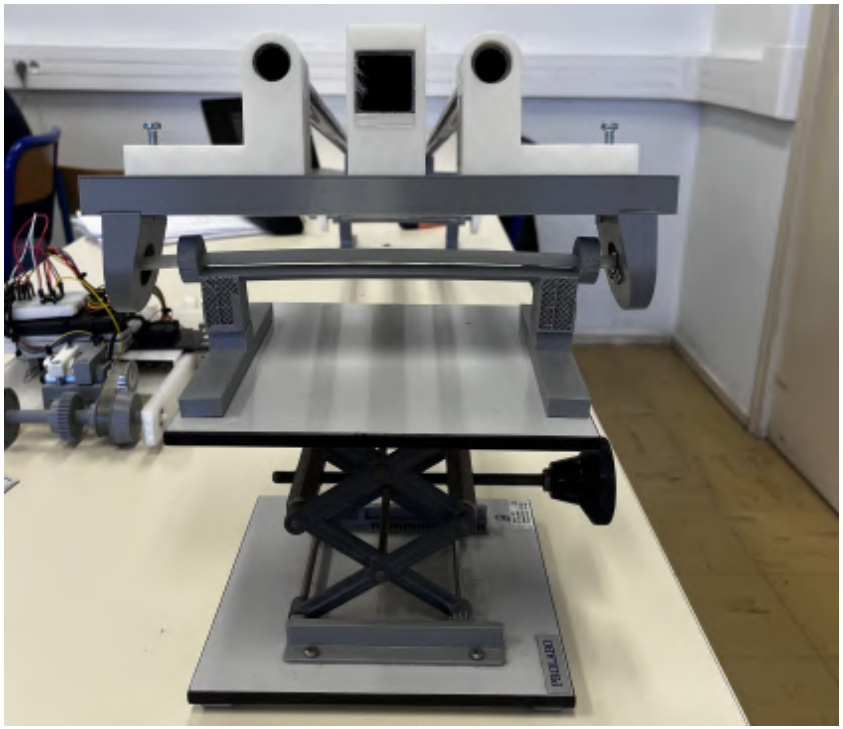
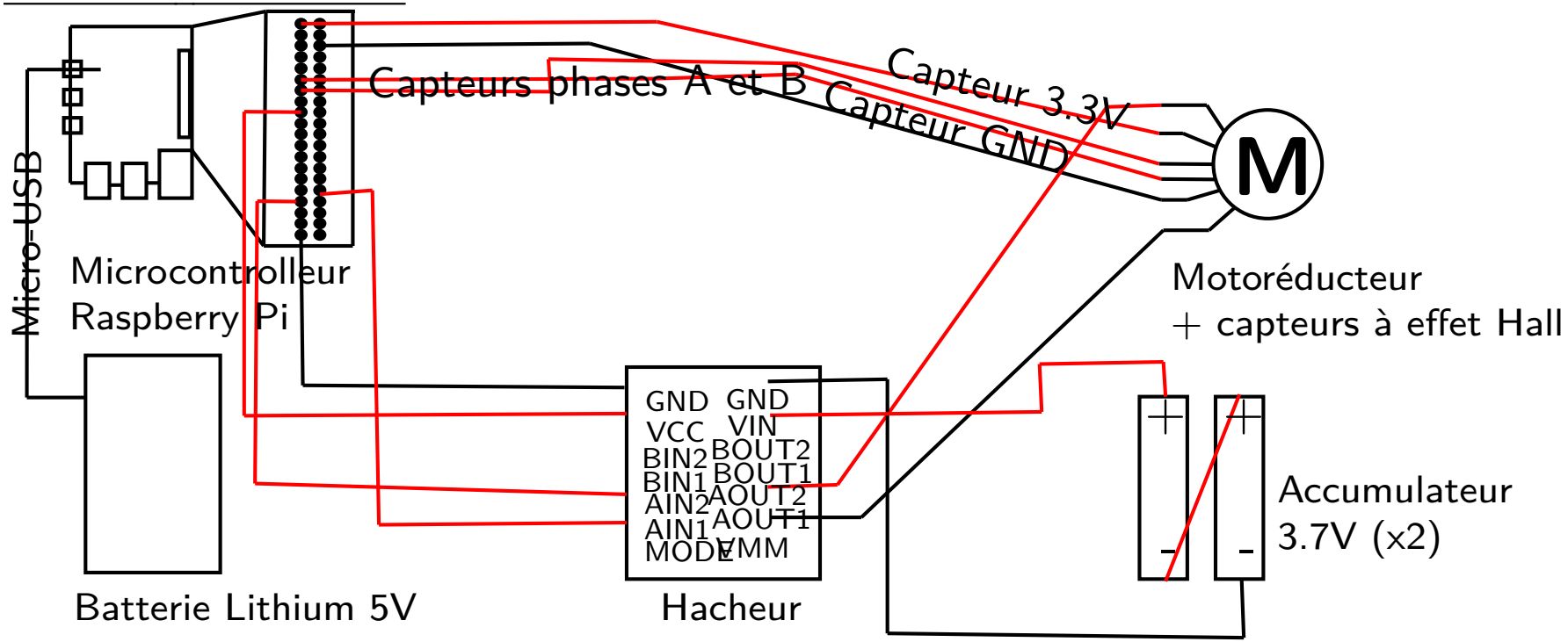


Schéma électrique :



Code d'asservissement en différents modes :

```python
## Importation des bibliothèques utiles :
import RPi.GPIO as GPIO
import time

## Configuration des pins GPIO pour les capteurs à effet hall et le hacheur :
hall1_pin = 17
hall2_pin = 27
GPIO.setmode(GPIO.BCM)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(12, GPIO.OUT) #Hacheur voie A (marche arrière)
GPIO.setup(hall1_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(hall2_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
pwm = GPIO.PWM(13, 1000) # Fréquence de 50 Hz
pwm_back = GPIO.PWM(12, 1000)
pwm.start(0) # Démarre le PWM avec un GPIO.setup(12, GPIO.OUT) #Hacheur voie A (marche arrière)rapport cyclique de 0%
pwm_back.start(0)

## Variables pour enregistrer la position et la vitesse actuelles :
position = 0
speed = 0

## Variables pour enregistrer le temps des fronts montants et descendants :
time1 = time.time()
time2 = time.time()

## Variables pour la consigne et le correcteur :
mode = "speed" # "position", "speed" ou "acceleration".
consigne = 1000 #consigne pour un asservissement en position.
consigne_vitesse = 2000 #consigne pour un asservissement en vitesse.
consigne_acceleration = 50000 #consigne pour un asservissement en accélération.

## Fonction de callback pour les fronts montants et descendants des capteurs à effet hall
## Fonction appelée lorsque le capteur Hall 2 change d'état
def hall2_callback(channel):
    global hall1_state, speed, position
    hall1_state = GPIO.input(channel)
    hall2_state = GPIO.input(channel)
    if GPIO.input(27) == 0:
        if GPIO.input(17) == 1: # rotation dans le sens horaire
            position -= 1
        else: # rotation dans le sens antihoraire
            position += 1
    else:
        hall1_state = GPIO.LOW

## Fonction appelée lorsque le capteur Hall 1 change d'état
def hall1_callback(channel):
    global hall2_state, speed, position
    hall1_state = GPIO.input(channel)
    hall2_state = GPIO.input(channel)
```

```python
    hall2_state = GPIO.input(channel)
    if GPIO.input(27) == 0:
        if GPIO.input(17) == 1: # rotation dans le sens horaire
            position += 1
        else: # rotation dans le sens antihoraire
            position -= 1
    else:
        hall2_state = GPIO.LOW

## Configuration des interruptiGPIO.setup(13, GPIO.OUT)ons pour les fronts montants et descendants :
GPIO.add_event_detect(hall1_pin, GPIO.BOTH, callback=hall1_callback)
GPIO.add_event_detect(hall2_pin, GPIO.BOTH, callback=hall2_callback)

## Espacement du temps entre chaque mesure et nouveau tour de boucle d'asservissement :
dt = 0.04 #en secondes

## Variables utiles à la fonction control :
erreur_prec = 0
erreur = 0



## Fonction de contrôle en boucle fermée : (contiendra les différents correcteurs selon l'asservissement souhaité)
def control(mode, consigne, Kp, position, speed, speed_old, speed_old_old, dt):
    #Introduction de variables globales utiles au programme :
    global integrale
    global erreur
    global derivee

    #Calcul de l'erreur selon le mode choisi :
    if mode == "position":
        error = consigne - position
    elif mode == "speed":
        erreur = consigne_vitesse - speed
    elif mode == "acceleration":
        erreur = consigne_acceleration - (speed - speed_old) / dt

    #Asservissement en position: Correcteur P (proportionnel):
    if mode == "position":
        Kp = 0.07 #gain proportionnel
        duty_cycle = Kp * error
        if duty_cycle >= 0:
            pwm_back.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 100) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, 0) # Limite le rapport cyclique à 0%
            pwm.ChangeDutyCycle(duty_cycle)
        else:
            pwm.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 0) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, -100) # Limite le rapport cyclique à 0%
            pwm_back.ChangeDutyCycle(-duty_cycle)
```

```
        return error

    #Asservissement en vitesse: Correcteur PID (proportionnel + intégrateur + dérivateur):
    if mode == "speed":
        Kp = 0.07 #gain proportionnel
        Ki = 0.1 #gain intégration
        Kd = 0.1 #gain dérivation
        erreur_prec = erreur

        #Calcul du terme intégral:
        integrale += erreur*dt

        #Calcul du terme dérivée:
        derivee = (erreur-erreur_prec)/dt

        #Calcul du rapport cyclique de sortie:
        duty_cycle = Kp*erreur + Ki * integrale + Kd*derivee

        #print("duty:" + str(duty_cycle))
        if duty_cycle >= 0:
            pwm_back.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 100) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, 0) # Limite le rapport cyclique à 0%
            pwm.ChangeDutyCycle(duty_cycle)
        else:
            pwm.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 0) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, -100) # Limite le rapport cyclique à 0%
            pwm_back.ChangeDutyCycle(-duty_cycle)
        return erreur
    #Correcteur PID (proportionnel + intégrateur + dérivateur):
    if mode == "acceleration":
        erreur_prec = (last_speed - last_last_speed)/dt

        #Calcul du terme intégral:
        integrale += erreur*dt

        #Calcul du terme dérivée:
        derivee = (erreur-erreur_prec)/dt

        #Kp, Ki, Kd
        Kpp = 0.0005
        Kii = 0
        Kdd = 0

        #Calcul du rapport cyclique de sortie:
        duty_cycle = Kpp*erreur + Kii * integrale + Kdd*derivee
        print("erreur : ", erreur)
        print("duty_cycle: ", duty_cycle)
        #print("duty:" + str(duty_cycle))
```

```python
        if duty_cycle >= 0:
            pwm_back.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 100) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, 0) # Limite le rapport cyclique à 0%
            pwm.ChangeDutyCycle(duty_cycle)
        else:
            pwm.ChangeDutyCycle(0)
            duty_cycle = min(duty_cycle, 0) # Limite le rapport cyclique à 100%
            duty_cycle = max(duty_cycle, -100) # Limite le rapport cyclique à 0%
            pwm_back.ChangeDutyCycle(-duty_cycle)

        return erreur

##Récupération des données :
liste_temps = [] #contiendra la liste les instants auquels sont prélevées les valeurs
liste_positions = [] #contiendra les valeurs associées à ces instants
debut = time.time() #enregistre l'instant relatif de début du programme

last_speed = 0
last_last_speed = 0

time.sleep(3)
print("début")

# Boucle principale pour exécuter le contrôle en boucle fermée
try:
    while time.time() - debut < 30:
        #récupération des données dans des listes:
        liste_temps.append(time.time() - debut)
        liste_positions.append(position)

        error = control(mode, consigne, Kp, position, speed, last_speed, last_last_speed, dt)
        #print(error, consigne, speed)
        last_position = position
        time.sleep(dt) # Attente de dt (en s)
        last_last_speed = last_speed
        last_speed = speed
        speed = (position - last_position)/dt
        #print("Vitesse cible:", consigne_vitesse, "Vitesse actuelle: ", -(last_position - position)/dt)
        acceleration = (speed - last_speed)/dt
        #print("Acceleration cible:", consigne_acceleration, "Acceleration actuelle: ", acceleration)

except KeyboardInterrupt:
    pwm.stop() # Arrête le PWM avant de quitter
    GPIO.cleanup() # Nettoyage des pins GPIO
    print(liste_positions)
    print(liste_temps)

pwm.stop() # Arrête le PWM avant de quitter
GPIO.cleanup() # Nettoyage des pins GPIO
print(liste_positions)
print(liste_temps)
```

Code de la dernière expérience :

```python
import RPi.GPIO as GPIO
import time

# Configuration des pins GPIO pour les capteurs à effet hall et le hacheur
hall1_pin = 17
hall2_pin = 27
GPIO.setmode(GPIO.BCM)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(12, GPIO.OUT) #Hacheur voie A (marche arrière)
GPIO.setup(hall1_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(hall2_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
pwm = GPIO.PWM(13, 1000) # Fréquence de 50 Hz
pwm_back = GPIO.PWM(12, 1000)
pwm.start(0) # Démarre le PWM avec un GPIO.setup(12, GPIO.OUT) #Hacheur voie A (marche arrière)rapport cyclique de 0%
pwm_back.start(0)

rapport_cyclique = 20 #On initialise le rapport_cyclique à une valeur suffisamment basse.
debut = time.time() #On prend une référence temporelle.

# Boucle principale pour exécuter le contrôle en boucle fermée
try:
    while time.time() - debut < 0.8:
        rapport_cyclique += 0.1 #On augmente de x à chauqe tour de boucle.
        time.sleep(0.4) #On temporise
        pwm.start(rapport_cyclique) #On applique la valeur de rapport cyclique itérée.

    rapport_cyclique = 0 #On augmente de x à chauqe tour de boucle.
    pwm.start(rapport_cyclique) #On applique la valeur de rapport cyclique itérée.

    #À partir de là, on joue avec la valeurs de rapport cyclique et de temps d'espacement des changements.
    rapport_cyclique = 1.5
    while time.time() - debut < 5:
        pwm_back.start(rapport_cyclique)
        rapport_cyclique += 0.3
        time.sleep(0.5) #Espacement des changements de rapport cyclique.


except KeyboardInterrupt:
    pwm.stop() # Arrête le PWM avant de quitter
    GPIO.cleanup() # Nettoyage des pins GPIO


pwm.stop() # Arrête le PWM avant de quitter
GPIO.cleanup() # Nettoyage des pins GPIO
```