



développement de reines connectées pour cavalier

résumé

J'ai réalisé mon stage au sein de l'université Paris Saclay l'intitulé du stage est le suivant :

Le respect du bien-être animal est une préoccupation de plus en plus présente pour les cavaliers modernes qui souhaitent pratiquer une équitation éthique. Pourtant, le cavalier doit utiliser des rênes qui exercent une pression sur l'équidé. Il devient donc primordial pour le cavalier d'utiliser ses rênes de manière à exercer une pression minimale pour communiquer ses intentions au cheval. Pour l'aider, nous nous proposons de développer des rênes intelligentes ("SmartReins") capables de mesurer en permanence la pression exercée et de l'enregistrer. Ceci permettra au cavalier d'analyser ses séances sur une application mobile afin de monter avec une plus grande légèreté dans ses mains. Également les rênes intégreront un dispositif de feedback permettant, si une trop grande pression est exercée, d'informer immédiatement le cavalier pour qu'il puisse adapter son équitation en temps réel.

1.Recherches préliminaires.....	2
1.1 Le poid à mesurer.....	2
1.2 Fonctions recherchées.....	3
1.3 Choix du matériel.....	3
2.Capteur de force.....	3
2.1 Choix du capteur.....	3
2.2 Branchements et calibrage.....	5
2.3 Correction de problèmes.....	6
3.Enregistrement et traitement des données.....	6
3.1 Stockage des données.....	6
3.2 Transmission des données.....	6
4.Prototypage et tests.....	7
4.1 Prototypage de l'objet.....	7
4.2 Tests.....	10
4.3 Prototypage de l'application.....	10
5.Conclusion et améliorations.....	11
6.Sources.....	12

1. Recherches préliminaires

1.1 Le poid à mesurer

Avant de pouvoir commencer le projet nous avons dû faire une première partie de recherche et d'expérience. En effet, dans un premier temps nous avons essayé de d'approximer le poids que nous allions devoir mesurer. Pour cela nous avons fait une première expérience :

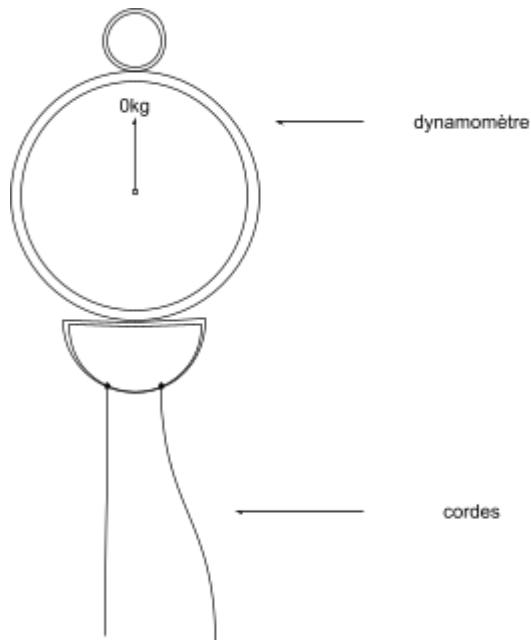


figure 1 : premier montage expérimentale

Le but de l'expérience est de calculer la force que les mains d'un cavalier vont avoir sur la bouche du cheval. Pour cela nous utilisons un dynamomètre pour représenter la bouche du cheval et deux cordes pour représenter les reines du cavalier. Pendant l'expérience, le prototype est accroché horizontalement sur une table par le haut du dynamomètre. Les cordes reposent sur la table sans tensions et le tarage du dynamomètre est effectué. Puis le cavalier prend les cordes en main et exerce les mêmes actions de main qu'il aurait sur un cheval et on note les valeurs observées.

actions fines	200 à 700 g
actions normales	1 à 2 kg
actions fortes	4 à 5 kg

Le principal problème de notre expérience est qu'elle a lieu à l'arrêt et dans le calme ce qui ne prend pas en compte les actions très fortes qui peuvent avoir lieu. Cependant nos mesures nous permettent de déterminer la précision dont nous allons avoir besoin pour notre capteur.

Pour déterminer le poids maximum que nous allons devoir mesurer nous nous sommes basés sur les travaux de Jean d'Orgeix dans son livre (1) dans lequel on peut lire :

"Le cavalier, se mettant en équilibre sur ses étriers, peut passer alternativement de 0 à 40 kilos de tension de rênes, d'une fraction de seconde à l'autre, sans aucun effort, sans perdre son aisance ni le contrôle de ses actions."

On peut voir que ses mesures sont bien supérieures au nôtre mais elles ont été faites en équilibre sur les étriers ce qui permet d'exercer une force beaucoup plus conséquente.

Après avoir fait nos expériences personnelles et avoir lu le livre de Jean d'Orgeix nous en avons conclu que nous allions devoir mesurer un poids entre 0 et 40 kilos avec une précision d'environ 200 grammes (pour détecter toutes les actions de main).

1.2 Fonctions recherchées

Maintenant que nous savons ce que nous allons devoir mesurer nous pouvons réfléchir aux fonctions que notre objet connecté va devoir embarquer.

- La première fonction évidente que nous voulons est de pouvoir mesurer le poids sous les conditions que nous avons décrites dans la partie précédente.
- Nous voulons aussi que notre objet nous prévienne de manière immédiate par un son si la tension des reines est trop élevée.
- L'objet doit aussi être capable d'enregistrer les mesures pour ensuite les transmettre au téléphone de l'utilisateur pour qu'il puisse analyser après la séance le poids qu'il exerce sur le cheval dans le but de s'améliorer
- Le point précédent implique aussi le développement d'une application mobile et l'utilisation d'un protocole pour communiquer les données à l'application.
- un bouton pour changer les modes (enregistrement de données, transfert de données).
- La taille de l'objet doit être minimal pour ne pas gêner le cavalier.

1.3 Choix du matériel

La première étape est de choisir le microcontrôleur que nous allons utiliser. Nous avons vu dans la partie précédente que l'objet doit être petit et doit pouvoir envoyer des données (soit par WIFI soit par Bluetooth). Nous avons donc choisi le Raspberry pi pico w qui répondait à toutes nos attentes. Pour le reste des composants nous avons utilisé des composants électroniques basiques sauf pour le capteur de force qui sera discuté dans la partie suivante.

2.Capteur de force

2.1 Choix du capteur

Le capteur est la partie essentielle du projet, en effet c'est le composant principal de l'objet (c'est lui qui va produire nos données). Le choix de ce capteur a donc été important.

Dans un premier temps nous allons étudier les différents types de capteur de force et comment ils fonctionnent, ici nous présenterons seulement des capteurs à jauge de contrainte, c'est la technologie la plus utilisée pour mesurer un poids.

Le capteur de force de traction : ce type de capteur permet de mesurer la tension d'un câble ou d'une corde par exemple.



figure 2 : capteur de force de traction

Ce type de capteur correspond à notre demande car en le plaçant sur les reines du cheval nous pourrions en calculer la force qu'exerce le cavalier dessus. Cependant ce type de capteur à un défaut, il est seulement utilisé dans l'industrie ce qui le rend assez cher. Il existe plusieurs types de capteur de traction comme le capteur en "s" par exemple mais l'utilisation reste la même.

Le capteur de force de compression : ce type de capteur permet de mesurer le poids d'un objet quand on le pose dessus il est utilisé dans les balances par exemple.



figure 3 : capteur de force de compression

Ce type de capteur ne correspond pas exactement à notre demande mais il a l'avantage d'être beaucoup moins cher. On peut noter que la plupart des capteurs de traction peuvent aussi être utilisés en compression mais l'inverse n'est pas vrai.

Ces deux types de capteurs utilisent des ponts de Wheaton, nous ne détaillerons pas comment marche exactement cette technologie car sa compression n'est pas cruciale pour le projet.

Après un peu de réflexion nous avons choisi de commencer le projet avec un capteur de force de compression car nous voulions dans un premier temps faire des tests sans avoir besoin d'investir dans un capteur à force de traction qui ne nous conviendrait peut-être pas. Nous avons choisi un capteur avec un poids maximum de 50 kg ce qui correspond à notre cahier des charges.



figure 4 : capteur de compression choisi

Ce type de capteur de compression est le plus répandu et il existe beaucoup de tutoriels pour apprendre à l'utiliser contrairement au capteur de traction.

2.2 Branchements et calibrage

La première étape une fois le capteur reçu à été de faire les branchements et de comprendre comment il marche. Un capteur de force ne peut pas se brancher directement sur un microcontrôleur, il lui faut une carte amplificatrice et la plus utilisée est la HX711.

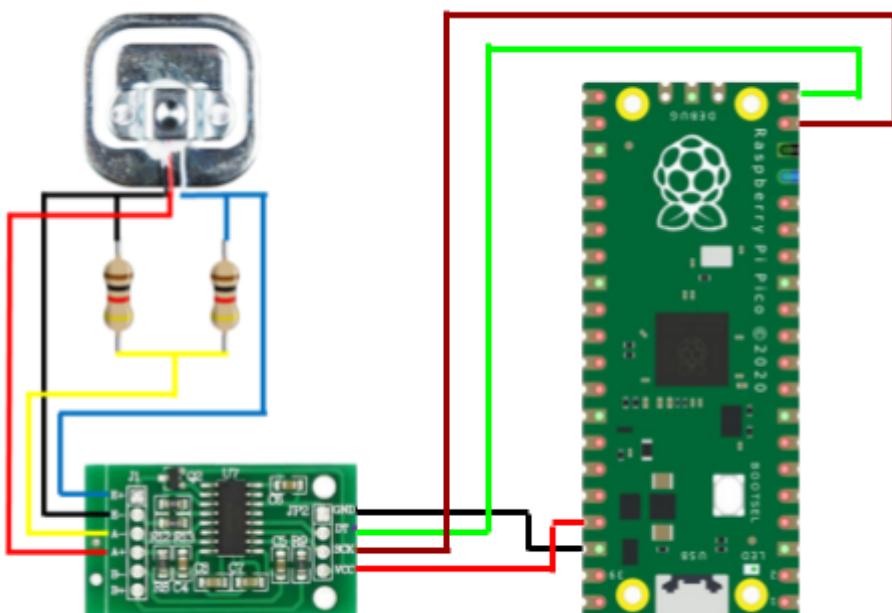


figure 5 : montage d'un capteur de compression sur le Raspberry pi pico w avec une carte HX711

Les capteurs de force à jauge de contrainte renvoient des valeurs en millivolts (mv) et ces valeurs doivent être converties en kilogrammes pour être compréhensible.

on a la formule $P = a * M + b$ avec $a, b \in \mathbb{R}^2$

Où M est la valeur mesurée en mv et P est le poids équivalent

On peut donc calculer a et b en faisant deux mesures, par exemple au repos (0 kg) et avec un poids prédéfini accroché au capteur ($i \text{ kg } \forall i \in [1, 50]$) puis en résolvant le système à deux équations on retrouve a et b.

2.3 Correction de problèmes

La théorie est assez facile à comprendre mais la pratique s'est avérée plus complexe car ce type de capteur est assez sensible et capricieux : en effet les valeurs du capteur varient beaucoup en fonction de l'environnement et de la position des fils. Pour corriger cela nous avons rajouté une phase de tarage à chaque initialisation pour que la valeur de départ soit bien à 0 kg en modifiant la valeur de b. Pour les problèmes de fils nous avons décidé de tout souder a la place de laisser tout sur la planche de test pour éviter les faux contacts.

Après cette étape nous avions un capteur fiable.

3. Enregistrement et traitement des données

3.1 Stockage des données

Maintenant que nous avons des données fiables nous pouvons les enregistrer, notre première idée a été d'enregistrer seulement les valeurs clés pour éviter de stocker trop d'informations, par exemple stocker la valeur maximum, la valeur minimum et la moyenne à chaque minute d'enregistrement.

Cependant après réflexion ce type de stockage n'était pas adapté pour notre problème. En effet, les actions de main sont en général très rapide et sont donc quasiment invisibles avec seulement des valeurs clés .Nous avons donc choisi d'enregistrer la valeur de chaque mesure dans un tableau dans un premier temps .Seulement nous avons eu un problème que nous redoutions, la taille de la mémoire du microcontrôleur était trop petite et nous n'arrivons qu'à stocker seulement 5 minutes d'enregistrement. Nous avons donc choisi d'enregistrer les valeurs dans un fichier une fois que la mémoire commence à se remplir.

3.2 Transmission des données

Nous avons maintenant des données enregistrées, nous devons les communiquer. Nous avons choisi d'utiliser le wifi avec le protocole suivant :

- le microcontrôleur se met en mode transfert de données , crée un point d'accès wifi et un serveur web
- l'utilisateur se connecte au wifi créé et va sur le serveur web
- le microcontrôleur envoie un fichier json avec toutes les données, arrête le point d'accès wifi et se met en mode enregistrement de données

Le point technique ici est l'envoi des données car comme dit précédemment les données ne tiennent pas dans la mémoire vive, il faut donc fabriquer le fichier json à la main en chargeant puis en envoyant les données en petit batch.

Nous sommes maintenant capables de communiquer les données.

4. Prototypage et tests

4.1 Prototypage de l'objet

Nous avons maintenant développé les principales fonctionnalités de notre objet. Nous pouvons maintenant commencer le prototypage. Cependant il reste un dernier point à régler. En effet, nous avons choisi un capteur de force de compression mais nous avons besoin d'un capteur de force de traction pour corriger cela, nous avons converti le capteur de la manière suivante :

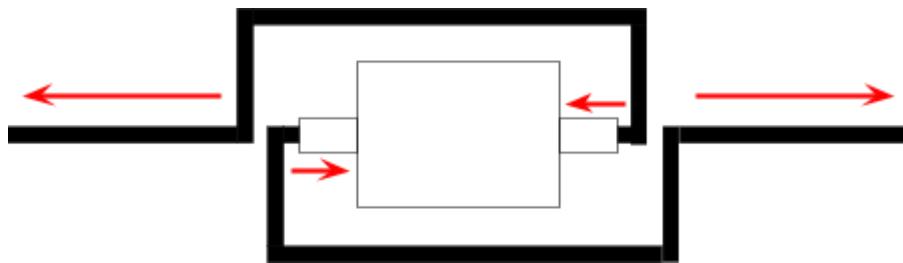
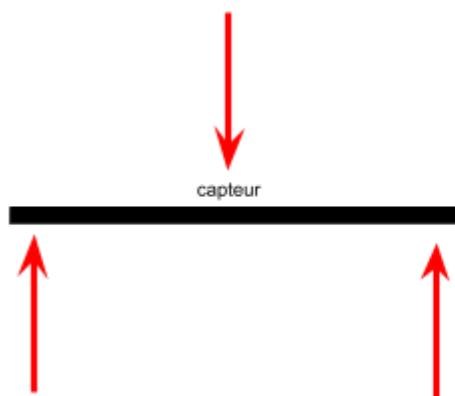
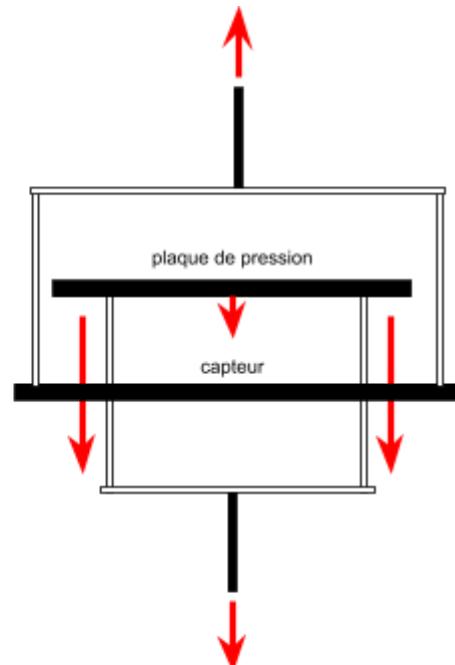


figure 5: conversion d'un capteur de compression en capteur de traction

avec notre capteur le schéma de la figure 5 peut se voir comme ceci :



capteur en mode compression



capteur en mode traction

figure 6 : montage de notre capteur selon le mode de mesure choisi

Sur la figure 6 en mode traction, la plaque de pression fait pression sur le capteur quand les deux bouts sont tendus ce qui provoque une compression et permet de faire la mesure.

Après cela nous avons bien un capteur de traction et nous pouvons commencer le prototypage.

Nous avons choisi de faire notre premier prototype à l'aide d'une imprimante 3D avec un filament en PLA et une buse de 0.4 mm. Pour le modèle 3D nous avons utilisé le logiciel Blender. Pour notre modèle nous voulions seulement une coque pour le capteur pour pouvoir continuer d'avoir accès au microcontrôleur et pouvoir brancher et tester les autres composants du projet.

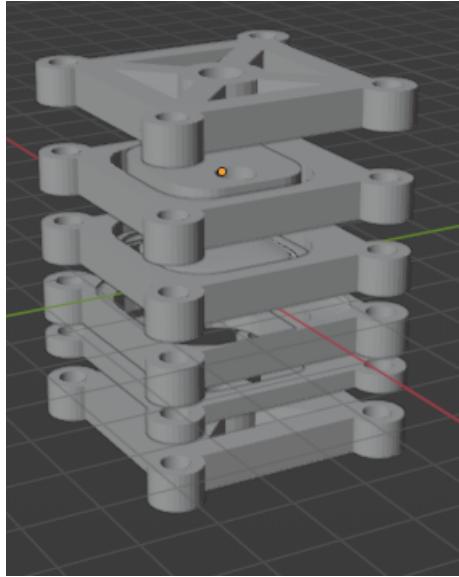


figure 7 : le prototype

Une fois le tout monté, nous avons ajouté un bouton pour pouvoir lancer le mode de transfert de données, un mini haut parleur (Buzzer) pour pouvoir faire un feedback immédiat à l'utilisateur sur le poids qu'il exerce avec ses mains. De plus, nous utilisons la led intégrée dans le microcontrôleur pour signaler à l'utilisateur dans quel mode il se trouve.

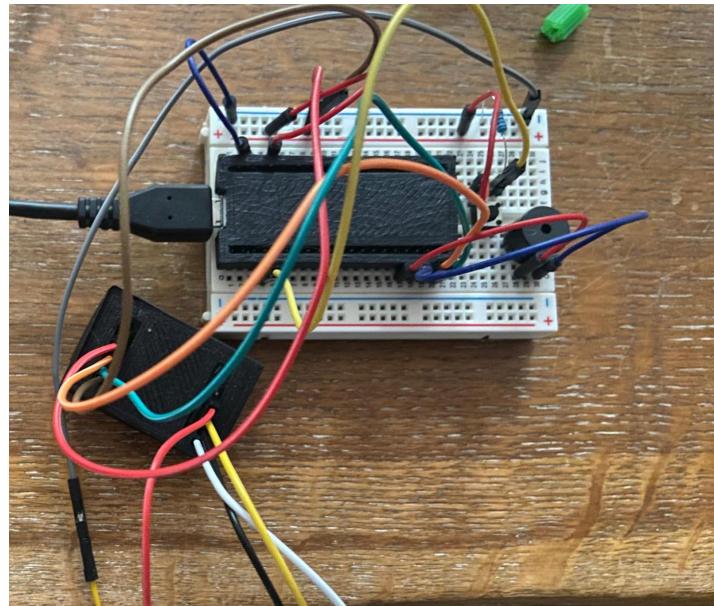


figure 8: montage expérimental

4.2 Tests

Une fois le montage terminé, nous avons commencé les tests. Pour faire cela nous avons calibré le capteur de force comme expliqué dans la partie dédiée. Puis nous avons repris le principe de la figure 1 en remplaçant le dynamomètre par notre prototype et nous avons obtenu les résultats suivants:

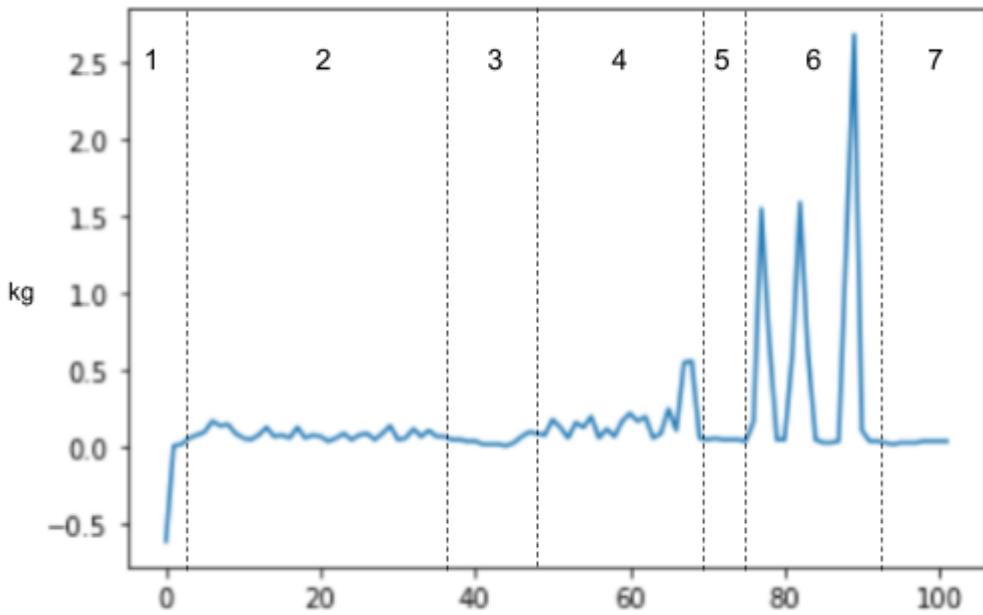


figure 9 : résultat de l'expérience

- (1) initialisation du capteur
- (2) actions de main légères
- (3) pause
- (4) actions de main normales
- (5) pause
- (6) actions de main fortes
- (7) fin des mesures

On peut bien distinguer les différentes périodes ce qui est très prometteur cependant nous avons des mesures plus faibles que durant notre première expérience, cela peut-être à cause de la différence entre les cordes utilisées. De plus le dynamomètre utilisé dans la première expérience était peu précis.

4.3 Prototypage de l'application

L'application mobile est le point que nous avons négligé, en effet nous ne voyons pas l'utilité de la développer tout de suite tant que nous n'avons pas fait les premiers tests en conditions réel avec des cavaliers différent car c'est à ce moment que nous allons voir les données qui sont importante

que nous voulons montrer à l'utilisateur pour qu'il puissent s'améliorer. Cependant pour pouvoir présenter notre projet nous avons décidé de faire un premier jet à l'aide de python et du module kivy.

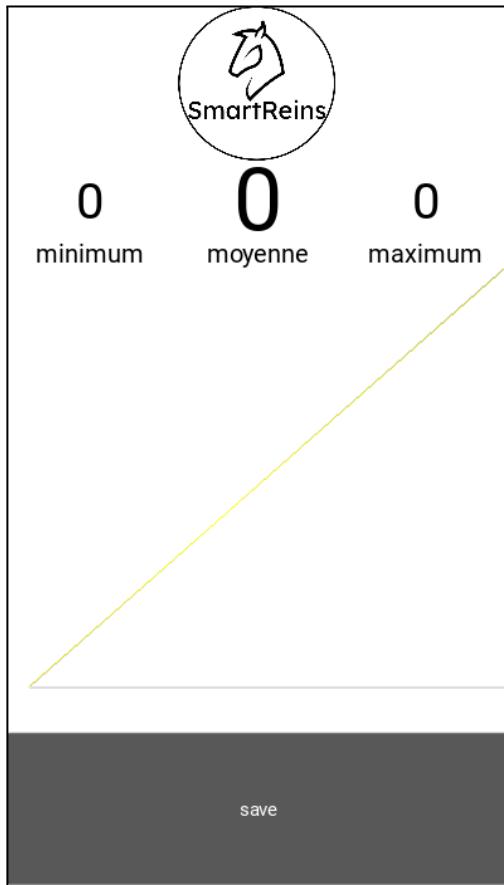


figure 10 : ébauche de l'application mobile

5. Conclusion et améliorations

Pour conclure, une grande partie de ce projet a été de faire de l'expérimentation avec le capteur et la conception du prototype. Les premiers résultats que nous avons eu nous semblent prometteurs. Nous devons maintenant améliorer notre prototype pour qu'il embarque en son sein toutes les composantes. Puis nous pouvons commencer la phase de test en condition réel avec plusieurs utilisateurs et avancer le développement de l'application.

Le principal problème de notre produit est actuellement la taille, en effet le bloc du capteur est plutôt imposant. C'est pour cela que nous avons décidé de continuer des tests avec un autre capteur qui cette fois-ci est un capteur de traction maintenant que les résultats nous semblent prometteurs.

6.Sources

(1) *L'équitation de saut d'obstacles 1/ analyse de la doctrine*, Jean d'Orgeix,
<http://excerpts.numilog.com/books/9782221000724.pdf>

(2) inspiration pour les branchements :

<https://www.instructables.com/Tutorial-to-Interface-HX711-With-Load-Cell-Straigh/>

(3) la carte HX711 avec micropython :

<https://github.com/SergeyPiskunov/micropython-hx711>