

Système d'illumination de trottoir nocturne pour piétons

* ELG 4539: Électronique III

1st Idriss Amadou Ali *Dept de Science NI:3001516551*

Université d'Ottawa

Ottawa, Ontario

iamad056@uottawa.ca 

2nd Marc-Antoine Mayer *Dept de Génie NI:300052042*

Université d'Ottawa

Ottawa, Ontario

mmaye047@uottawa.ca

Abstract—Nous avons construits un dispositif capable de différencier les entités sur le trottoir basé sur des capteurs de mouvement infra-rouge et des capteurs de poids, tous reliés à un microcontrôleur Arduino, qui effectuera un CAN sur les signaux analogiques et un Raspberry pi 4 qui effectue une prise de décision basée sur le flux de données reçues.

Index Terms—Raspberry Pi, Arduino, Capteur de Poids, Capteur Infrarouge, Django

I. INTRODUCTION

Le gouvernement du Québec définit la pollution lumineuse comme toute lumière projetée vers le ciel qui obstrue l'observation des étoiles [1]. Cet effet est dû à la projection et la réflexion de la lumière vers le ciel ainsi qu'une surabondance de lumière. Ce phénomène représente un manque d'efficacité dans le déploiement de la lumière. D'un point de vue environnementaliste, c'est aussi un gaspillage d'énergie puisque l'énergie lumineuse qui s'échappe vers le ciel ne contribue pas à la tâche voulue d'un lampadaire, qui est par exemple d'illuminer une rue ou un trottoir.

Les capteurs infrarouges pyroélectriques (PIR) comptent parmi la classe de capteurs détecteurs thermiques [2]. Ces capteurs mesurent la radiation incidente grâce au changement de leur température. Lorsqu'on présente au détecteur un certain matériel absorbant, on peut le configurer pour répondre à une certaine plage de fréquences. Les PIR ont été conçus principalement pour la détection des corps humains, ce qui veut dire que les longueurs d'onde désirées sont de huit à douze micromètres.

II. CONCEPTION

A. Conception Matérielle

Pour le hardware du projet, nous avons décidé de sélectionner deux types de capteurs : le HX711 et le SR-501. Le capteur de tension HX711 est composé d'un Wheatstone bridge, ce circuit utilise 4 résistances arrangeées comme dans

la figure ci-dessous. Le voltage de sortie mesurée varie selon la tension mécanique appliquée sur le membre du capteur. Il suffit par la suite qu'à calibrer ce voltage de sortie à l'aide un poids ou une force connue. Le capteur a une limite de poids de 20 kilogrammes, mais dans le cas des simulations pour ce projet, ce n'est pas un problème.

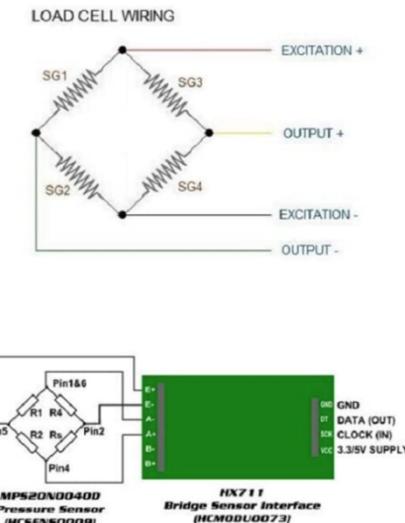


Fig. 1. Capteur de Poids HX711. [3]

Quant au capteurs SR-501, ils fonctionnent en détectant la radiation infra-rouge et en donnant une valeur de sortie binaire, soit 0 ou 1, dépendamment du rapport entre la radiation détectée et un certain seuil. Ces capteurs ont une limitation sur leur temps d'activation. Lorsque la sortie d'un capteur passe de 1 à 0, elle sera prise en mode LOW pour 2 secondes. Cela veut dire que la sortie du capteur sera fixe à 0 pour deux secondes peu importe la radiation mesurée. Ceci prouve être une contrainte pour la conception des cas de test.

capteur de poids. Le schéma ci-dessous représente le raccordement des capteurs infrarouges à un Arduino. Le logiciel utiliser pour ce schéma ne comprenait pas le capteur HX711 dans sa base de données et l'Arduino Nano non plus. Le HX711 as 4 pins, comme vu dans la figure plus haute, où une pin est pour l'alimentation 5 volts, une autre pour la terre, une troisième pour la pin d'entré digitale du Arduino et la dernière pour l'horloge. [4]

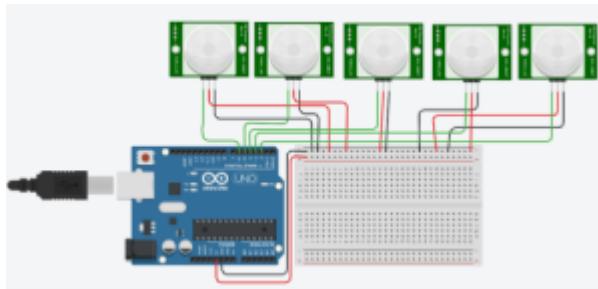


Fig. 2. Intégration de capteurs IR Hc-Sr501. [5]

B. Conception Logicielle

Afin d'effectuer l'acquisition de données, une synergie entre l'environnement python et C++ arduino est nécessaire. L'état du système sera transféré au raspberry pi qui effectuera de calculs pour déterminer l'état du système. Cet état sera mis dans un format Json transférable à un ract dashboard en ligne via protocole https.

1) *Code C++:* Ce code sera téléversé sur l'arduino Nano via l'application platformio IDE. Le microcontrôleur est de type ATmega328P/Pa et cette configuration est nécessaire afin de téléverser le code C++ écrit dans le fichier "main.cpp". Ce fichier prend en charge l'initialisation des capteurs ainsi que la mesure de leurs valeurs et les broadcast continuellement à une baudrate de 76800 envoi pour réception à baudrate de 19800. le rafraîchissement de la boucle principale est 1kHz.

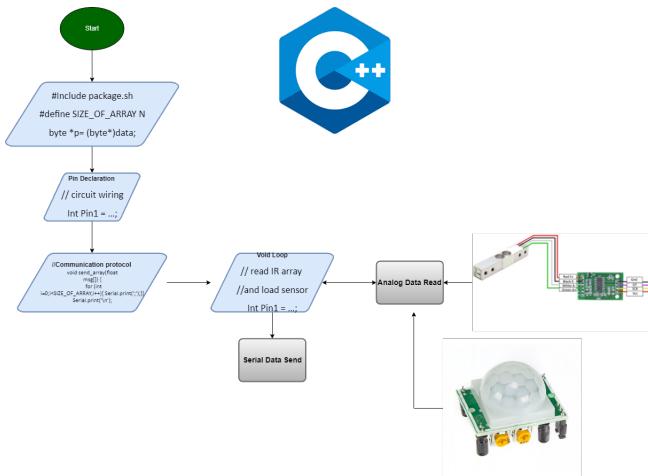


Fig. 3. organigramme du Code C++

Pendant l'exécution de la boucle principale l'arduino enverra une chaîne de dimension 2 et contenu 8bits , une flottante

et un entier. La flottante est le poids capté sur le trottoir et l'entier est un encodage des position activées de la chaîne de capteur infrarouge. Le plus simple format pour encoder l'entier est l'encodage binaire, cela est fait en multipliant la valeur des capteurs par une puissance de 2 de leur position, pour 5 capteurs en commençant à 0:

$$\left\{ \begin{array}{l} V_i \in \{0,1\} \\ N = \sum_{i=0}^4 (V_i \cdot 2^i) \end{array} \right. \quad (1)$$

Cette encodage à l'avantage d'ajouter l'information de la position si la route envisagée est à sens unique et on assume un mouvement dans une direction. L'entier N est la deuxième entrée de la chaîne binare de 8 bits envoyée par l'arduino.

2) *Code Python:* Le code python assurera l'intermédiaire entre le dashboard et les capteurs, il décidera l'état du système à afficher grâce à la matrice d'état suivante:

	Poids léger	Poids moyen	Poids lourd
Vitesse lente			
Vitesse vite			
Vitesse très vite			

Fig. 4. Matrice d'états du système

Afin de lire les valeurs des capteurs, le module Serial va détecter l'arduino et se connecter à son port, on tire avantage du parallélisme offert par le langage informatique python pour définir l'état et l'envoyer à la même vitesse que l'on lit les valeurs des capteurs grâce au module Thread. Une thread va agir en mode daemon(arrière plan) et se terminer lorsque la valeur booléenne définissant la lecture deviendra 0 ou FALSE.

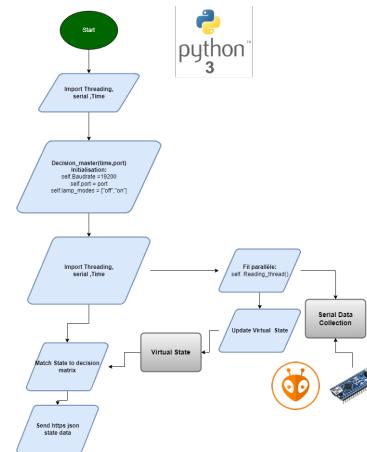


Fig. 5. Organigramme du code Python

Le code va calculer la vitesse à partir de la position. Pour décoder la position à partir de N on va utiliser le logarithme base 2 en combinaison de la fonction partie entière. Les capteurs

infrarouges sont espacées de $d = 15\text{cm}$ et donc la vitesse revient à diviser la différence de position multipliée par d , N étant l'entier lu due l'arduino:

$$\begin{cases} N = \sum_{i=0}^4 (V_i \cdot 2^i) \\ P = \lfloor \log_2(N) \rfloor \\ v = \frac{dP}{dt} = \frac{\Delta P \cdot d}{\Delta t} \end{cases} \quad (2)$$

Le module python time utilisé dans le fil parallèle permet de traquer le temps et effectuer ces mesures.

L'état du système est stocké dans un format json prêt à être envoyé dans une requête https.

3) Déploiement Django: Le backend django permet le déploiement d'un serveur capable de recevoir les requêtes https, le frontend est un tableau de bord écrits en React javascript. Le code du projet est disponible sur Github pour consultation



Fig. 6. Tableau de bord react

[6]

III. MISE EN OEUVRE

Afin de compléter le projet le développement matériel et logiciel a été effectué en parallèle puis joint quand leur développement fut suffisamment accomplis. Le schéma suivant est donc obtenu:

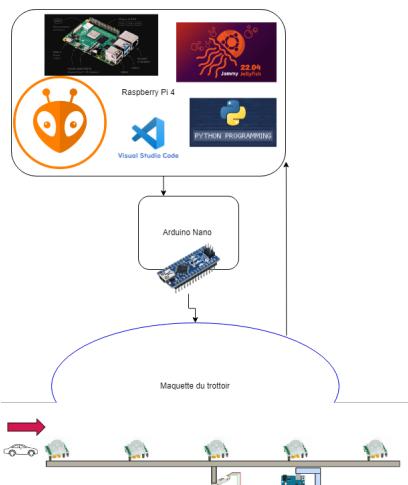


Fig. 7. Schéma du trottoir

Durant la démonstration des séries de test furent effectués afin de tester les limites de notre design. Le poids et la vitesse ont été ajusté à l'échelle de la maquette de trottoir.

Dû au mismatch des baudrate le raspberry pi ne pu être utilisé et un laptop fut utilisé à sa place.

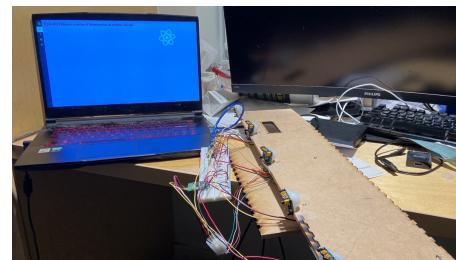


Fig. 8. Maquette Test

A. Tests

1) Test1 objet immobile: On pose un objet immobile, le résultat est l'affichage de son poids, le tableau de bord affiche un lampadaire éteint. Le test est un succès.

2) Test2 objet léger à mouvement rapide: On fait bouger un objet sur la maquette de trottoir, le poids reste le même mais le lampadaire ne s'allume pas, le statut n'affiche pas la valeur voulue, piéton. Le test est un échec.

B. Budget Final

TABLE I
BUDGET

number	Component	Price(CAD)
1	HX711 Load Cell	15,99
1	HC-SR501 IR sensorx5	18,99
1	Nano Board ATmega328Px3	40,89
Total		75,87

IV. DISCUSSION

Le travail effectué lors de ce projet est fonctionne comme une preuve de concept pour le système d'éclairage de trottoir nocturne pour piétons mais plusieurs lacunes sont à noter:

- Problème de synchronisation de la baudrate entre émetteur et lecteur prévenant l'utilisation du Raspberry Pi 4 comme ordinateur miniature
- Problèmes de synchronisation au niveau de la lecture des données serialisées et mise à jour de l'état du système en format Json
- La plateforme de réception de requête https au niveau du backend Django est incomplète.
- Le dashboard n'a pas encore de fonction callback capable de rafraîchir les données en temps réel(1kHz) sans recharger la page entière

Une majeure limitation à laquelle nous avons fait face lors de l'implémentation des cas de test est le cône de détection des capteurs infrarouges. Les capteurs SR-501 ont un certain champ de vision qui a la forme d'un cône. Cela veut dire que la portée efficace des capteurs dépend de la distance entre ceux-ci. Si les capteurs sont trop proches, leurs zones de détection sont superposées. Cela réduit la précision du array de capteurs quand ça vient à la dérivation de la position des occupants de la rue. Les capteurs peuvent détecter un objet avant qu'ils soient

exactement en avant du capteur. Pour résoudre cela, il faudrait concevoir des corridors pour chaque capteur afin de limiter leur champ de vision, un peu comme œillères pour chevaux. D'un point de vue d'implémentation réelle, cette solution n'est pas aussi pratique que pour la simulation en laboratoire. Toutefois, dans l'application réelle, la distance entre les capteurs serait plus grande donc la superposition des champs de vision est un problème moins important.

- [4] "How HC-SR501 PIR Sensor Works and How To Interface It With Arduino," Last Minute Engineers, Jul. 03, 2018. <https://lastminuteengineers.com/pir-sensor-arduino-tutorial/> (accessed Dec. 07, 2022).
- [5] "Infrared Sensors Overview: Types, Functioning and Use Cases — Kisi". <https://www.getkisi.com/guides/infrared-sensors> (consulté le 10 octobre 2022).
- [6] https://github.com/Driss-001/ELG4539_trottoir(accessed on 8th December 2022)

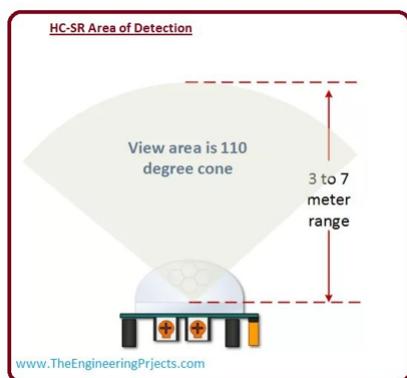


Fig. 9. Cône de détection du capteur IR

V. CONCLUSION

Nous avons réussi à établir les fondations d'un système de contrôle d'illumination de trottoir sélectif basé sur la vitesse et le poids d'un objet à sens unique. Les problèmes de détections proviennent surtout du chevauchage du cône de portée des capteurs Infrarouge causé par leur proximité, ce qui réduit l'efficacité de leur implémentation à l'échelle de test présente. Le tableau de bord synchronisé avec la matrice de décision affiche bien l'état du système mais la continuité de la mise à jour du document json doit faire l'effet d'un suivi temporel pyroélectriques afin d'éviter les retards et une mise à jour effective de l'état du système sur le tableau de bord. Plus de développement est nécessaire afin d'aboutir à un format final.

RECONNAISSANCE

Nous voudrions remercier le professeur Mahmoud Youssef pour l'opportunité d'accomplir ce projet ainsi que l'assistant à l'enseignement Mohammed Yassine Bouhamidi pour son aide et orientation dans l'implémentation des technologies de communication en ligne.

REFERENCES

- [1] "La Pollution Lumineuse." Les Avantures De Rafale, Environnement Et Lutte Contre Les Changements Climatiques, Québec, <https://www.environnement.gouv.qc.ca/jeunesse/chronique/2005/0503-causes.htm>.
- [2] Zappi, Piero, et al. "Tracking Motion Direction and Distance with Pyroelectric IR Sensors." IEEE Sensors Journal, vol. 10, no. 9, Sept. 2010, pp. 1486–1494, <https://doi.org/10.1109/jsen.2009.2039792>.
- [3] "4-Wire Load Cell (1/5/10/20/200kg) with HX711 and Arduino - Circuit Journal". <https://circuitjournal.com/four-wire-load-cell-with-HX711> (consulté le 8 octobre 2022).