Rodrigues Hugo

Neyret Anthony

Rapport de Projet Tutoré

Robot détecteur de Gaz

Sommaire :

* Cahier des charges
* Schéma Général
* Schéma de câblage
* I) Module
* A) Déplacement
* 1) Moteur
* 2)Shield moteur
* 3)Démultiplexeur
* B) Communication
* C) Capteurs
* 1) Capteur CO
* 2) Capteur de la zone de stockage
* 3) Capteur de contact
* II) Conception de la carte d’alimentation
* III) Difficultés rencontrées
* IV)Points à améliorer
* V) Conclusion
* VI) Annexes

Cahier des charges

* Robot mobile ayant pour but de trouver l’origine d’un gaz sur un terrain connu et de le signaler à l’utilisateur par mail.
* Alimenté par batterie.
* Système Wi-Fi.
* Capteur de Gaz.
* Suis un chemin prédéfini par une bande noire au sol.
* Camera

Schéma Général

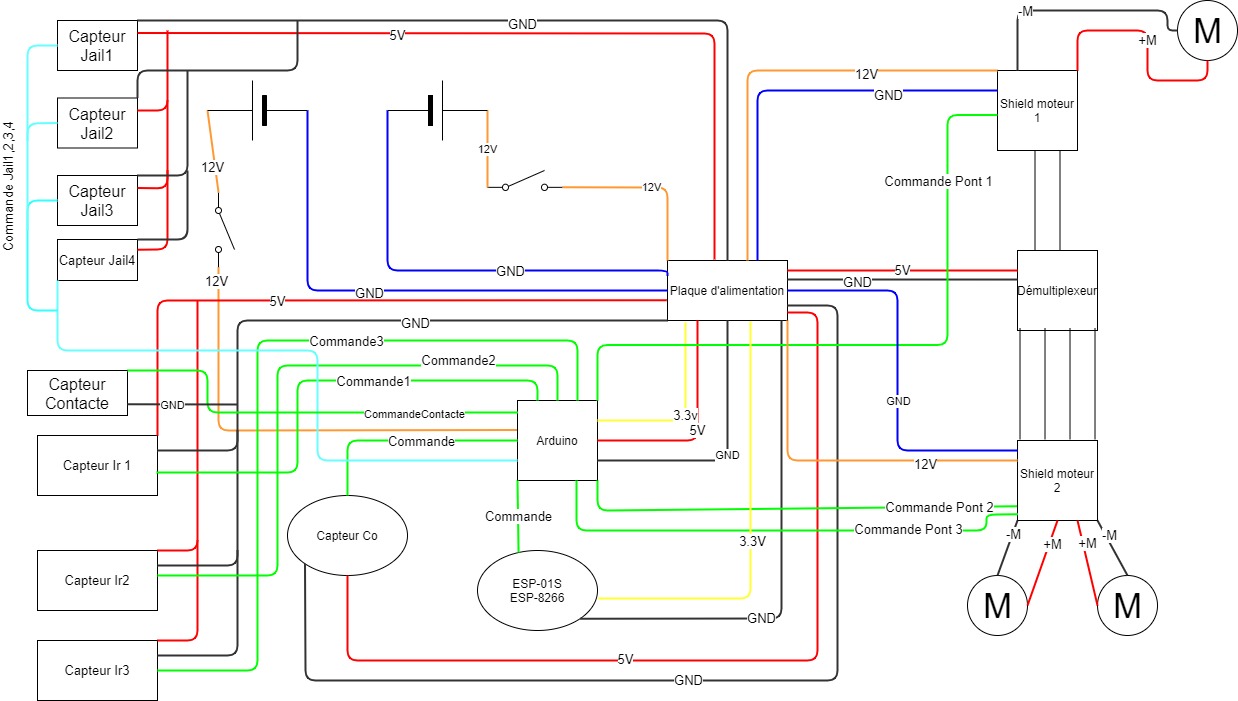


Schéma de câblage

I)Module

A) Déplacement :

Pour respecter le cahier des charges, et l’utilisation des moteurs à courant continu fournit avec le robot. On a choisi de fixer des Shields moteur et des capteurs infra-rouge Arduino au-devant du robot pour détecter la ligne noir. Pour choisir les bons cas pour suivre la ligne on a dressé un tableau logique des entrées et sorties :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Entrées |  |  | Sorties |  |  |  |  |  |
| Capteur  Gauche | Capteur Centre | Capteur  Droit | D | G | Ad | Ag | Stop | Av |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | x | x | x | x | x | x |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | x | x | x | x | x | x |

Les 0 représentes les fois où les capteur IR détecte la ligne.

1)Moteur :

Les moteur reçus avec le Robot sont des GR42X25,24V.

Un courant nominal de 0.86A.

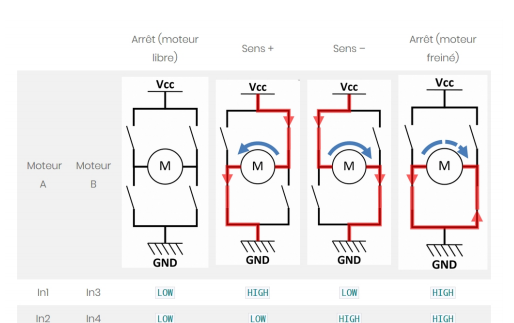
Puissance nominale de 14.3W.

Tension nominale de 24 V.

Nous avant fait le choix d’alimenter les moteurs en 12V via les Shield piloté PMW par Arduino pour

2) Les Shields :

Pour piloter les moteurs, on a dû ajouter 2 Shields L298H qui intègre un double pont en H. Les Shields sont alimenter par les batteries via la plaque d’alimentation en 12 V. L’activation d’un pont en H (gestion de la vitesse des moteurs) est pilotée par l’Arduino. Le sens de rotation des moteurs est quant à lui géré par le démultiplexeur. On justifie le choix d’avoir pris des Shields par le fait que l’Arduino ne pouvait pas alimenter seule les moteurs (sortie Arduino 5V et 40mA).



C) Démultiplexeur :

Lors de la conception du plan on sait rendu compte que nous allions manquer de pin PMW sur l’Arduino pour piloter les 2 Shields après réflexions et recherche sur internet on à décider d’opter pour un démultiplexeur 74HC595N.

Alimenter par la plaque d’alimentation en 5V (broche 16 et 8 masse).

Les sorties sont sur les broches de 1 à 7 et la broche 15 (Qn).

L’entrée des données série, qui va commander les sorties du composant, se trouve sur la broche 14 (serial data input)

Sur la broche 10 on trouve le Master Reset, pour mettre à zéro toutes les sorties. Elle est active à l’état BAS. Alors on forcer à un état logique HAUT, en la reliant par un +5V de la plaque de l’alimentation.

La broche 13, output enable input, est une broche de sélection qui permet d’inhiber les sorties. En cela signifie que lorsque cette broche n’a pas l’état logique requis, les sorties du 74HC595 ne seront pas utilisables. Nous l’avons reliée à la masse.

La n°11 et la n°12. Ce sont des "horloges" Lorsque nous envoyons un ordre au 74HC595, nous envoyons cet ordre sous forme d’états logiques qui se suivent. Par exemple l’ordre 01100011. Cet ordre est composé de 8 états logiques, ou bits, et forme un octet (une suite de 8 bits). Cet ordre va précisément définir l’état de sortie de chacune des sorties du 74HC595. Le problème c’est que ce composant ne peut pas dissocier chaque bit qui arrive.

C’est là qu’intervient le **signal d’horloge.** Ce signal est en fait là dans l’unique but de dire si c’est un nouveau bit qui arrive, puisque le 74HC595 n’est pas capable de le voir tout seul. En fait, c’est très simple, l’horloge est un signal carré fixé à une certaine fréquence. À chaque front montant (quand le signal d’horloge passe du niveau 0 au niveau 1), le 74HC595 saura que sur son entrée, c’est un nouveau bit qui arrive.

B) Communication :

Le module communication a étais confectionnais grâce à un ESP8266-01S, qu’on à programmer via les liaisons RX TX de l’Arduino. L’ESP8266-01S est active par l’Arduino sur ça pin IO2. Lorsque l’ESP8266-01S est activé il envoie un mail avec sujet et texte pour signaler que le robot à détecter du gaz.

Il est alimenté en 3.3V sur sa pin VCC et EN et la masse sur sa pin GND.

C) Capteur :

Quand on à reçus le robot il était équipe de deux capteurs différents :

-capteur contacte.

-capteur infra-rouge.

On lui a rajouté un capteur de gaz.

Les capteurs infra-rouges sont utilisés pour détecter la zone de stockage du robot. Le capteur contacte quant à lui est là pour détecter un possible obstacle sur le chemin tracé.

1) Capteurs CO :

Pour répondre à la deuxième partie du cahier des charges ont acheté un capteur de gaz le MQ-2. On a décidé de prendre ce capteur car ça gamme de détection nous offre des choix plutôt varier (voir annexe). Le choix de la détection du capteur, il a deux pins différents pour le câblage sois une digitale sois une analogique et une tension d’alimentions de 5V ce qui nous évité de prendre une alimentation en plus pour le faire fonctionner on a fait le choix de prendre le pin analogique ce qui nous a permis de définir un seuil de détection ce qui revenais à être plus précis pour détecter le gaz.

2) Capteur de la zone de stockage :

Pour utiliser les capteurs infra-rouges fournies, on a décidé de crée une zone de stockage pour que le robot vienne stationner lorsqu’il à finit son tour de ronde.

Le capteur est alimenté en 5V et relié à la masse sur la plaque d’alimentation.

Le pin de commande est relié au pin digital de l’Arduino.

Le capteur fonction de sorte que quand il détecte la zone à 10 centimètres (zone minimale de détection) il s’arrête.

3) Capteur de contact :

Le capteur d’obstacle est une bande avec un contacteur ON/OFF.

Il est branché sur un pin digital de l’Arduino et sur la masse de la plaque d’alimentation.

Il fonctionne de telle sorte que quand il y a un objet sur le chemin du robot, il rentre en contacte avec le capteur et les moteurs du robot s’arrête.

II) Conception de la carte d’alimentation :

On à décider de déporte les alimentations (3.3V, 5V et masse) de l’Arduino sur une carte d’alimentation, ce qui nous permit de pouvoir alimenter les différents capteurs et de séparer le circuit commande du circuit puissance. La carte permet aussi d’alimenter les Shield moteur et l Arduino en 12 V (par la Pin Vin).

III) Difficultés rencontrées :

La camera nous a posé un problème, on sait aperçut que pour faire fonctionner la caméra Open MV H7 avec l’Arduino il fallait un Shield wifi qui permettait avec le code et le processeur de la caméra d’envoyer un flux vidéo à l’Arduino. Nous nous sommons pencher sur la caméra trop tard car le Shield couté assez cher à la commande et le délai de livraison aurait été trop long. Pour régler ce problème on opter 2 solutions que l’on à tester sans conclusion utilisable pour le projet.

La première solution consister à utiliser notre module ESP01-S brancher sur l’Arduino et faire communiqué les 2 mais la camera ne permettais pas de faire cette liaison sans son Shield.

La deuxième solution était d’acheté une caméra Arduino OV7670, mais là encore pas un tuto pour expliquer le fonctionnement, la librairie ardafruit OV7670 n'est pas compatible avec un Arduino UNO (c'est pour un µcontroleur de gamme SADM et pas ATM32 comme l'UNO).

Donc les deux solutions n’étant pas viables on a donc opter d’utiliser notre module ESP-01S pour envoyer un mail en cas de détection avec le capteur MQ2.

IV)Points à améliorer

Pour ce qui est des points d’amélioration, au niveau hardware,

Acheter le Shield Wifi OpenMV H7 pour faire fonctionner la caméra, changer les batteries en batterie rechargeable et crée une station de recharge dans la zone de stockage,

Au niveau software,

Ajouter dans le code de la zone de stockage un horodatage pour décider de la mise en route du robot et la fin de son cycle, modifier le code du capteur d’obstacle pour que le robot contourne l’objet et continu ça mission, dans l’ESP-01S envoyer un mail pour dire qu’il y a un obstacle à enlever, envoyer aussi un mail pour dire que le robot est dans sa zone de stockage et qu’il charge.

V) Conclusion :

Pour conclure sur ce projet de fin d’année, il nous à permit de pouvoir comprendre les démarches d’un projet sur plusieurs mois, gestion des contraintes de livraison et de compatibilité des fournisseurs que l’on peut rencontrer dans une entreprise.

Sa nous permit aussi la recherche de solutions pour répondre au cahier des charges quand la première ne fonctionner pas.

De travailler sur des contrôleurs que nous n’utilisons pas souvent l’Arduino UNO et l’ESP-01S.

Il nous aussi permit de développé des connaissances basic en réseau avec le protocole

VI) Annexes