UNIVERSITE PARIS SACLAY
INSTITUT D'OPTIQUE GRADUATE SCHOOL

Procédés de Traitement de l'Information et du Signal

Vision industrielle - livrable technique

Yacine El Yamani
Yiaho Yuan
Antoine Bourhis

Encadrants: Julien Villemajane

PALAISEAU, APRIL 2023

Contents

1 Vidéo de présentation

Pour observer le protoype en focntionnement suivre le lien : Vidéo

2 Introduction

L'objectif de ce projet est de construire un système de tri d'objets parcourant un tapis roulant selon leur couleur.

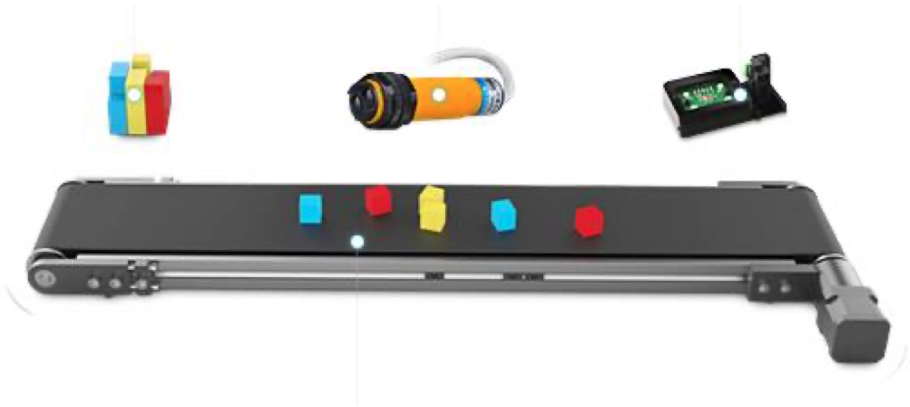


Figure 1: Schéma du prototype présentant le capteur de couleur, les cubes, et le convoyeur.

Dans ce projet, nous utiliserons un capteur de couleur et deux types de moteurs : 3 servo moteurs et un moteur pas à pas. Un microcontrôleur (carte nucléo) va contrôler la vitesse et le sens de rotation des moteurs et de l'autre le capteur qui permettra d'identifier et de filtrer les couleurs des blocs. Le capteur sera placé dans une boite dont l'éclairage se fera par une LED blanche et qui sera fixé au tapis de telle sorte que le capteur de couleur soit au dessus des cubes, au centre du convoyeur et que l'éclairage des cubes soit toujours le même.

Le schéma suivant montre l'approche logique de ce dispositif

2.1 Cahier des charges et performances attendues

Le système doit respecter le cahier des charges suivant :

- **Rapidité**: analyser au minimum 10 pièces par minutes.
- **Détection**: différencier les 4 couleurs de base suivantes : rouge, vert, jaune et bleu.
- **Fiabilité**: Une erreur d'une pièce sur 1000 est tolérée sur la détection des couleurs de base.
- **Ergonomie**: Une interface Humain Machine, permettant de transmettre la couleur (ou forme) des pièces à trier, pourra être développée.

2.2 Notice d'utilisation

Afin d'utiliser correctement le prototype, il suffit de poser les cubes un par un sur le conviyeur. Une fois la tension allumé, celui ci se met en route et le tric se fait par lui même. Il faut cependant attendre que l'objet à trier ait été éjécté du tapis avant de pouvoir mettre le cube suivant.

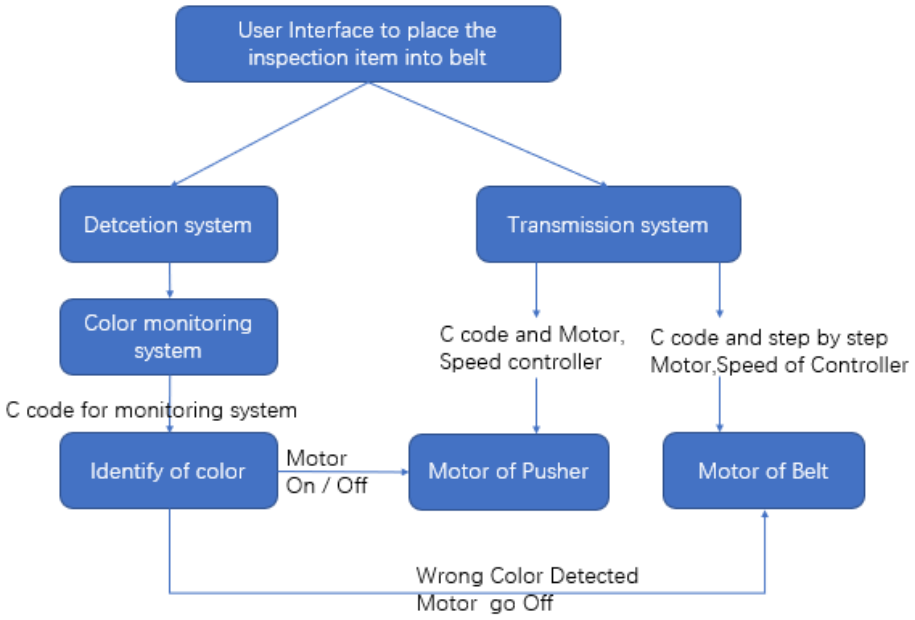


Figure 2: Diagramme logique de contrôle

2.3 Objectifs recherchés à travers ce projet

Ce projet sera un succès si nous parvenons à faire fonctionner le prototype et que chacun des membres du groupe ait participé de manière équitable et efficacement au succès du projet. Un autre challenge est celui de la communication qui se fait principalement en anglais puisqu'un membre du groupe est étudiant international. Le succès reposera donc aussi sur la compréhension par tous des problèmes rencontrées et des solutions apportées.

3 Description du projet

3.1 Matériel éléctronique

3.1.1 Moteur du convoyeur

Principe de fonctionnement : Le moteur du trolley est un moteur pas à pas bipolaire et donc nécessitant une inversion de courant. Il faut donc pour cela utiliser un **pont en H**. Pour cela on utilise donc deux cartes : L297 et L298 dont leur fonctionnement sont rappelés ici : http://lense.institutoptique.fr/mine/carte-dextension-l297/. l'alimentation du moteur est aux alentours de 5V et sa fréquence de rotation autour de 500 $\pm$ 100 Hz. La fréquence est à adapter en fontion de ce que souhaite l'utilisateur.

Branchements : de nombreux branchements sont à faire dans cette partie. Pour la carte L297, les beanchements à faire sont les suivants : 5V, Reset, CW, Vref et Enable doivent être branché à une tension d'alimentation de 5V. Le paramètre CLOCK doit être branché à la carte nucléo (ex : D8) où le signal codé sur MBED sera envoyé. ENABLE doit aussi être branché à la carte (Digital Input / Entrée permettant de valider le fonctionnement des sorties du composant et donc d'activer la mise en rotation du moteur). Enfin Half doit être connecté à la masse. Si on veut changer le sens de rotation du moteur, il faut brancher CW sur 0 ou 5V.

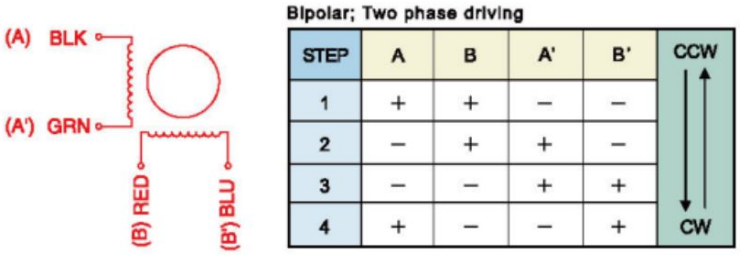


Figure 3: Principe de fonctionnemnt d'un moteur bipolaire. Schéma trouvé sur le site du fabriquant du moteur

Pour la carte L298 il y a 7 branchements à faire. Premièrement connecter les deux cartes entre elles. Ensuite il faut brancher V_{mot} à 5V et la masse à celle de la masse commune. Enfin les branchements au moteur du trolley se fait de la manière suivante : B2-1 ; B1-2 ; A2-4 ; A1-3. Voir photos à l'appui si nécessaire.

3.1.2 Capteur de couleur

Figure 4: Capteur de couleur

Principe de fonctionnement : Ce capteur de couleur fonctionne avec une commande au format I2C dont l'adresse est 0x10. Le format I2C utilise deux fils pour la transmission des données : un fil de données (SDA) et un fil d'horloge (SCL). Le fil de données transporte les informations binaires entre les périphériques connectés, tandis que le fil d'horloge fournit une synchronisation entre les périphériques en indiquant le moment où les données sont valides.Pour chaque périphérique connecté au bus I2C, celui ci possède une adresse unique qui lui permet d'être identifié par les autres périphériques où ces adresses sont codées sur 7 bits. Pour coder avec un detecteur au format I2C il faut:

- Commencer par définir l'adresse du capteur
- Initialiser le bus I2C en appelant la fonction

- Lire ensuite les valeurs données par le capteur à l'aide d'une requete read puis write sur le bit selectionné

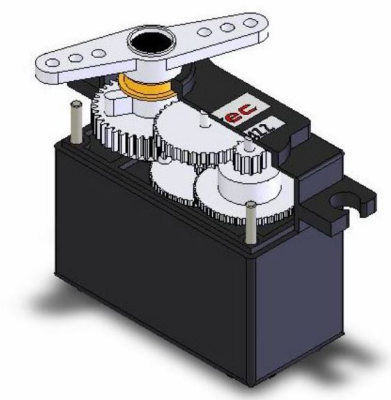
Le dispose aussi d'une RGB diode qu'il faut coupler à une LED blanche afin de s'assurer que la lumière amabiante n'influe pas sur les informations transmises par le capteur. C'est aussi cette LED de couleur blanche qui va nous permettre de faire une calibration des couleurs perçues. Les coefficients sont obtenus en suivant la balance des blancs, vu en cours de colorimétire de H.Sauer. En effet nous utilisons une propriétés colorimétrique du blanc ; les coordonnées colorimétriques (X,Y,Z) du blanc verifient $X = Y = Z = 0,33$. Il suffit donc juste d'éclairer le capteur à l'aide de la diode blanche et d'affecter un coefficient à chaque mesure (R,V et B) pour les rapporter à 0,33. On vérifie les données reçu en effectuant plusieurs mesures différentes de couleurs avec le lecteur de donnée Terraterm à l'aide d'une simple commande printf. Nos mesures nous ont données 4 seuils de detections un pour chaque couleur:

- Pour le rouge il faut R>0,45.
- Pour le vert V>0,4 et R<0,3
- Pour le bleu B>0,36
- Pour le jaune V>0,4 et R>0,38

Branchements : Le capteur doit être branché avec une tension de **3.3V**. Les informations du capteur sont transmises à la carte nucléo en reliant les bornes **SCL** et **SDA** entre les deux composants. Il faut cependant rajouter des résistances entre les deux composants afin de ne pas les abimer. On aura utilisé des résistances de 10 kΩ. Et enfin ne pas oublier de brancher la masse, noté **GND**.

Le code utilisé est donné en annexe. La documentation fabriquant est donné ici : https://www.mikroe.com/color-10-click.

3.1.3 Moteur poussoir



(a) Moteur Parralax utilisé

Voltage Specifications

Pin	Name	Description	Min	Typical	Max	Units
1 (white)	Signal	Input, TTL or CMOS	3.3	5.0	Vservo+0.2	V
2 (red)	Vservo	Power Supply	4.0	5.0	6.0	V
3 (black)	Ground	Ground		0		V

(b) Aide au branchement pour le moteur poussoir

Principe de fonctionnement : Les moteurs utilisés sont de la marque *PARALLAX* et sont des servos motors. Leur pilotage se fait par un signal PWM, vu en première année. Toute leur utilisation est rappelé sur le site du lense : http://lense.institutoptique.fr/nucleo-controler-un-mouvement-angulaire-2/.

Branchements : les branchements sont simples : Alimentation à 5V, masse à la masse commune et enfin la commande à la carte nucléo (D10,D11 ou D12 par exemple).

3.2 Objectifs et réalisation des fonctions

3.2.1 Moteur du convoyeur

Dans un premier temps il a fallu trouver les caractéristiques de fonctionnement du moteur pas à pas. Les test sont fait à l'oscilloscope en essayant plusieurs valeurs de tensions et de fréquence. Afin de vérifier que les les branchements on bien été effectués, on vérifie avec à l'oscillosocpe que les signaux ayant besoin d'être alimentés le sont.

On a décidé pour la réalisation du projet que les objets à trier devait s'arrêter en face des poussoirs. Il fallait donc pouvoir arrêter et faire fonctionner le moteur. D'où la nécessité d'avoir deux fonctions programmées. Le code en annexe permet de faire fontionner le motueur à la carte nucléo.

3.2.2 Capteur de couleur

Pour le capteur de couleur, il a fallu en premier lieu traduire les informations. Enfin pour son bon fonctionnement et que la lumière ambiante n'impacte pas la lecture de la couleur, une structure en bois a été construite afin de pouvoir placer le capteur au centre du convoyeur, éclairé par un LED blanche. La LED est alimenté par une tension de 5V et protégé par une résistance de 10 kΩ.

3.2.3 Moteur poussoir

De la même façon que pour le moteur pas à pas, les servo motors ont d'abord été caractérisé par oscilloscope afin de connâitre leur fonctionnement. Ensuite, une fois que la calibration a été effectué, on connaît le nombre de tours à faire pour que les poussoirs parcourent toutes les amplitudes, afin d'être sûr ques les cubes soit éjéctés du chariot. Il faut donc la aussi programmer deux fonctions. La première étant une fonction "pousser" et l'autre "tirer". Afin de controler la direction de rotation du servomoteur pour soit pousser soit tirer il suffit simplement de controler le temps haut du signal affecté au moteur.
D'après nos mesures de calibration un temps haut de:

- 600 µs est necessaire pour tirer le poussoir pour ne pas gener le trajet du cub
- 2000 µs est necessaire pour pousser le poussoir jusqu'au bord du tapis

Lorsque qu'un cube est détecté, celui-ci est poussé et ensuite le poussoir doit revenir à sa postition d'origine afin de ne pas bloquer les cubes suivants. De plus afin de vérfier leur bon fonctionnement on rajoute au préalable que les poussoirs soient bloqués dans une position initale étant celle où ils sont au bord du convoyeur. Le code est en annexe.

3.3 Intégration des modules entre eux

Une fois que tout les modules aient été étudiés séparément, il s'agit ensuite de les connecter ensmeble. Un première raison debug est l'alimentation des modules. Il faut faire attention à séparer l'alimentation du moteur pas à pas de celui qui connecte à la nucléo. Il faut donc avoir une alimentation externe qui aliment les moteurs

pas à pas et servo moteurs.

Ensuite il faut mesurer le temps entre la détection de couleur par le capteur et le moment où le cube arrive face au moteur poussoir qui doit l'éjecter du tapis. Une fois cela effectué, il suffit d'adapter le code des éléments pris séparément et les connecter entre eux. Le code se construit simplement en 4 parties:

- On mesure la vitesse v du tapis
- On mesure la distance d entre le capteur de couleur et chaque poussoir
- On affecte à chaque poussoir une couleurs
- Lorsqu'une couleur est détectée il suffit de faire avancer le bloc sur une durée $\tau = \frac{d}{v}$ puis d'arrêter le tapis et enclencher le poussoir.

Le code utilisé est donné en annexe.

4 Bilan

4.1 Avancement final

Nous allons maintenant comparer notre réalisation au cahier des charges que nous nous étions fixé :

Rapidité : Pour la rapidité, nous devions être capable de trier 10 cubes par minutes, malheureusement nous n'avons pu qu'en trier au maximum que 8 par minutes. Cela est dû au fait que notre algorithme ne trie les cubes qu'un par un. Il ne garde pas en mémoire les informations permettant de metttre plusieur cubes sur le convoyeur. Ce qui limite grandement la rapidité de notre systeme.

Afin de palier à ce problème notre programme doit rentrer à chaque detection d'une couleur(Rouge,Vert,Bleu,Jaune) le temps $(\tau_r, \tau_v, \tau_b, \tau_j,)$ necessaire au cube pour aller jusqu'au poussoir dans une liste. Celle-ci devra être triée en permanence et il faudra seulement faire avancer le bloc d'une durée t donnée par le premier element de la liste puis supprimer cet élément après le tri.

Détection : Pour la detection, nous avons reussi à trouver des valeurs seuils de R,V,B permettant à chaque couleur de bien être differenciée par le capteur.

Fiabilité Pour la fiabilité une erreur d'une pièce sur 1000 était tolérée , bien evidemment nous n'avons pas fait tourner notre programme pendant 1H40 néanmoins sur un temps bien plus court nous n'avons vu aucune erreur de tri lorsque le programme etait lancé.

Ergonomie La création d'une petite boîte support pour le capteur rendait beaucoup plus pratique la détection. Pour les branchements, il aurait été possible de souder les fils sur une plaquette permettant ainsi de faciliter les branchements à chaque séance. Enfin, par manque de temps, la partie interface de notre projet n'a pu être traitée.

4.2 Retour d'équipe et diagramme de Gantt

Le calendrier prévisonnel que nous avions à été très bien respecté puisque nous étions la plupart du temps dans les échéances que nous nous étions fixé. Les deux dernières séances n'ont pu être aussi productive qu'espéré, donc l'algorithme utilisé n'a pu être optimisé afin de respecter l'ensemble du cahier des charges que nous nous étions fixés.

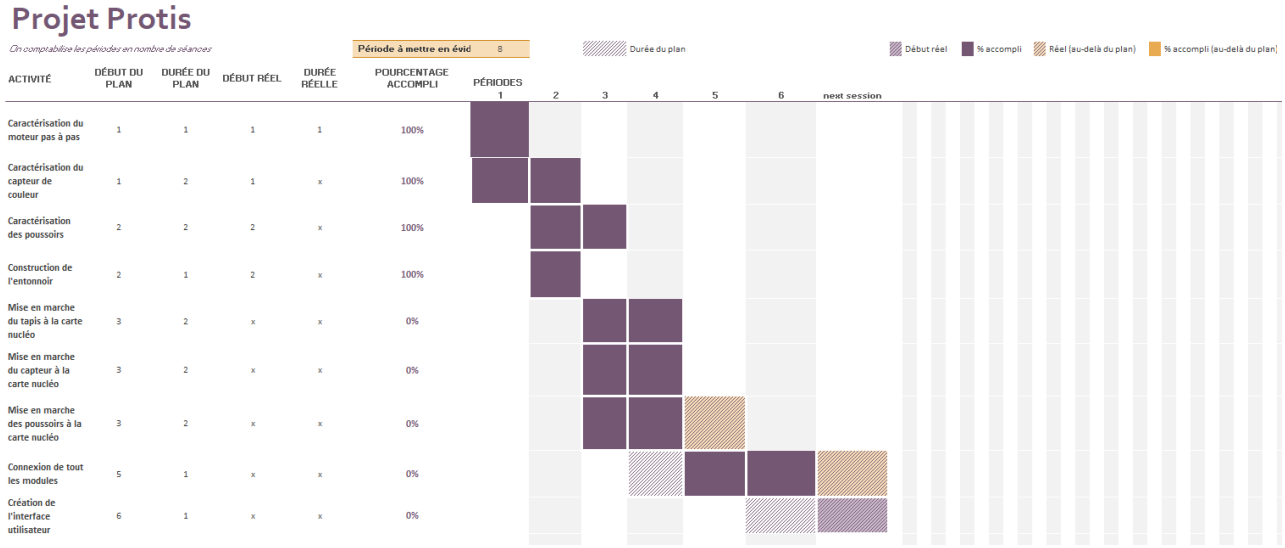


Figure 6: Diagramme de Gantt à la dernière séance

4.3 Difficultées techniques

Bien que le projet soit un succès, les sessions alloués pour travailler dessus on été rythmées par quelques difficultés tehniques. En voici quelques unes:

- **Branchements** : Les branchements était nombreux sur ce projet du fait notamment des différents modules nécessitant d'être tous raccordés entre eux. Plusieurs fois noous avons du refaire les branchements qui nous ont fait perdre du temps. De plus un mauvais branchement prenait beaucoup de temps avant d'être résolu.
- **Capteur de couleur** La dificulté du capteur de couleur résidant dans son principe de fonctionnement I2C. Ce principe était nouveau et il fallait donc se l'appropier totalement avant de pouvoir exploiter les données transmises.
- **Carte L297 et L298** : Ces deux cartes, ont au début posé un porblème pour comprendre leur fonctionnement, notamment par les nombreux branchements qu'il fallait effectuer. Une fois prise en main, leur fonctionnement était alors simple.

4.4 Conclusion génerale

Ce projet nous a permis de developper notre travail d'équipe en anglais en incluant tous les membres du groupe. Nous avions pour objectif que chacun comprenne l'ensemble des parties du projet, ce qui a été bien respecté. Notre objectif général à donc été realisé et nous sommes fiers d'avoir pu réussir à comprendre le fonctionnement de notre materiel qui n'etait pas forcement facile à prendre en main au début (Protocole I2C, utilisation de la carte L297 et L298 pour controler le moteur) et d'avoir pu en ressortir un prototype fonctionnel qui respecte en grande partie le cahier des charges.

5 Annexe

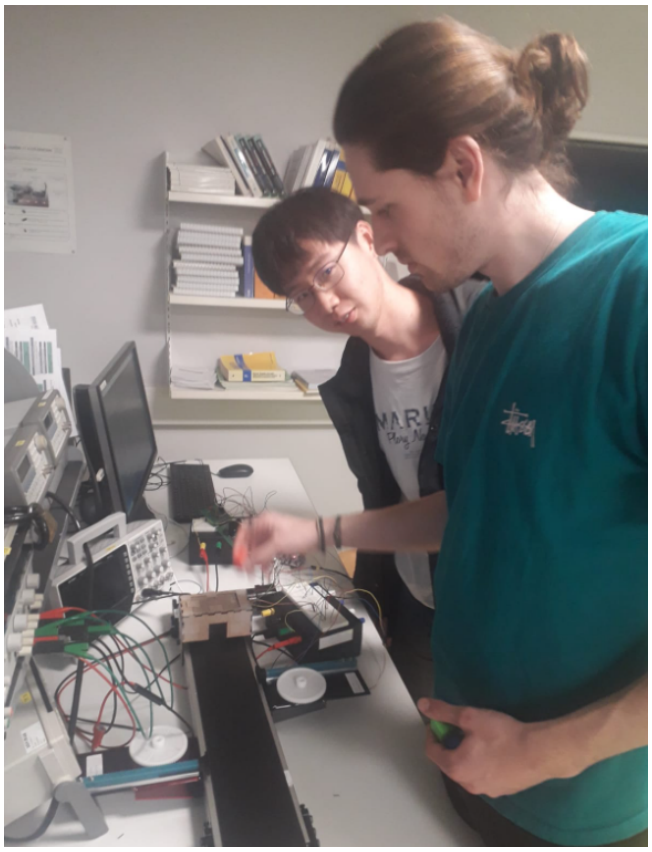


Figure 7: L'équipe en train de travailler

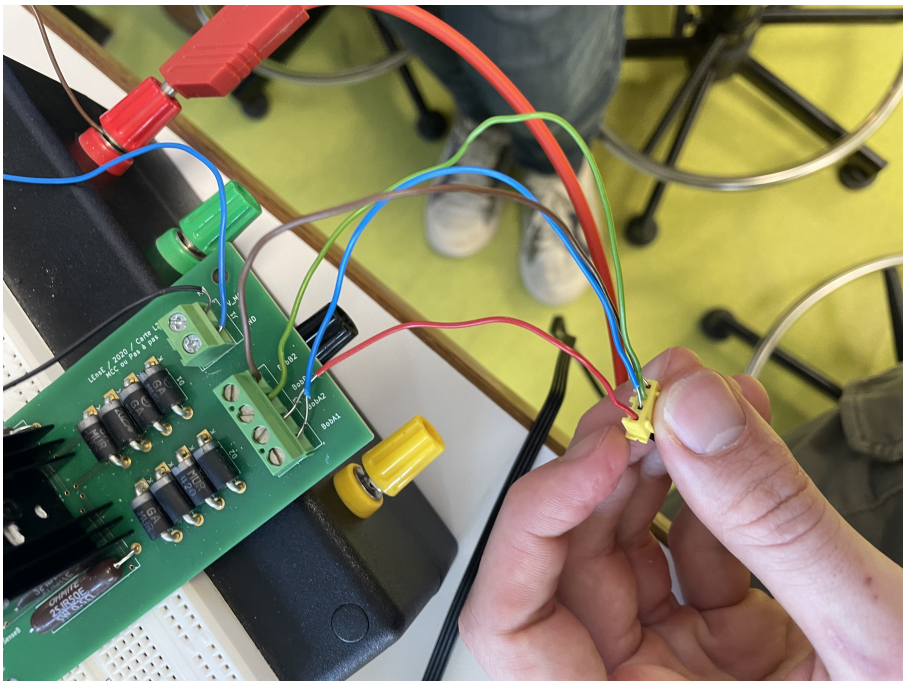


Figure 8: Branchement moteur pas à pas

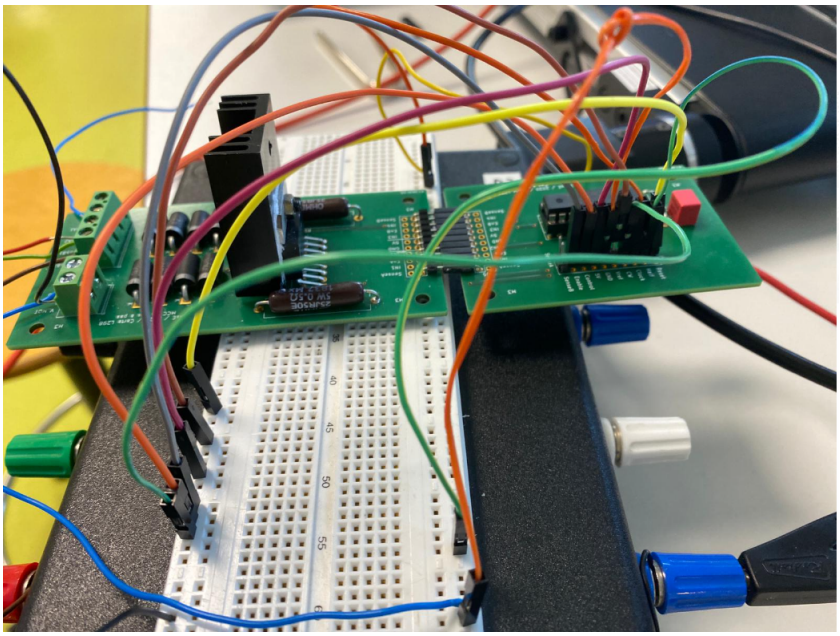


Figure 9: Branchement pont en H

```
void servomot_on() {
    servo_mot.period_us(2500);La periode est très importante elle permet de définir
la vitesse du trolley néanmoins l'intervalle de fonctionnement du moteur
se situe entre des periodes de 2000us a 2600us
    servo_mot.pulsewidth_us(1500); le temps permet d'allumer et d'eteindre le moteur
 1500-> allumé
}

void servomot_off() {

    servo_mot.period_us(2500);
    servo_mot.pulsewidth_us(0); 0->éteint
    }
```

Figure 10: Code MBed pour faire fonctionner le moteur pas à pas

```
On change la durée de temps haut pour choisir si l'on pousse ou si l'on tire
le poussoir
void pousser1() {
        poussoir1.pulsewidth_us(2000); 2000 pour pousser
         thread_sleep_for(1000); attendre 1 sec
       }

void tirer1() {
        poussoir1.pulsewidth_us(600);   600 pour tirer
        thread_sleep_for(1000);
```

Figure 11: Code MBed pour faire fonctionner les poussoirs

```
void capteur_color(double *Px,double *Py,double *Pz) utilisation de pointeurs pour
prendre plusieurs informations en sortie de la fonction
{
char    r[2];
char    v[2];
char    b[2];
char    cmd[3];

    cmd[0]    = 0x30;
    cmd[1]    = 0x00;
    cmd[2]    = 0x00;

    int k = my_i2c.write( 0x10 << 1, cmd, 3 );  démarrage du capteur à l'aide
    d'une requête I2C

    double R, V, B,X,Y,Z;

        cmd[0]    = 0x05;  // Rouge

        my_i2c.write( 0x10 << 1, cmd, 1, true );  // requete

        my_i2c.read( 0x10 << 1, r, 2 );      // reponse

        cmd[0]    = 0x06;  // vert

        my_i2c.write( 0x10 << 1, cmd, 1, true );  // requete

        my_i2c.read( 0x10 << 1, v, 2 );      // reponse
        cmd[0]    = 0x07;  // bleu

        my_i2c.write( 0x10 << 1, cmd, 1, true );  // requete

        my_i2c.read( 0x10 << 1, b, 2 );      // reponse
        R=r[0]+255*r[1];  Transformation en coordonnée colorimétrique
        V=v[0]+255*v[1];
        B=b[0]+255*b[1];
        X=(33.0/24)*R/(R+V+B); Calibrage des coordonnées pour que la somme
        Y=(33.0/46)*V/(R+V+B); en face d'une lumière blanche soit 1
        Z=(33.0/30)*B/(R+V+B);
        *Px=X;
        *Py=Y;
        *Pz=Z;
}
```

Figure 12: Code MBed pour faire fonctionner le capteur de couleur

```
// Projet Protis, S8 IOGS, portant sur la visison industrielle réalisé par
// Yacine El Yamani, Yiaho Yuan et Antoine Bourhis.


#include "mbed.h"
#include "ma_bibli.h"  // Dans la bibli on retrouve les fonctions tirer, pousser,
                       //  servomot_on, servomot_off

PwmOut servo_mot(D6);                 // Connexion du moteur tapis à la nucléo
PwmOut poussoir1(D10);                // Connexion du poussoir 1 à la nucléo
PwmOut poussoir2(D11);                // Connexion du poussoir 1 à la nucléo
PwmOut poussoir3(D12);                // Connexion du poussoir 1 à la nucléo

I2C     my_i2c(D14, D15);             // Connexion du capteur à la nucléo


int main(){

    int i=1;
    double X,Y,Z;
    servo_mot.period_us(2500);        // Mise en marche du moteur du conveyeur
    servo_mot.pulsewidth_us(1500);

poussoir1.period_ms(20);   // Permet de bloquer les poussoirs dans leur position d'origine.
poussoir1.pulsewidth_us(600);
poussoir2.period_ms(20);
poussoir2.pulsewidth_us(600);


my_i2c.frequency(400000);

    capteur_color(&X,&Y,&Z); // (X : coordonnées du rouge | Y : coordonnées du Vert
                             // | Z : coordonnées du Bleu)
```

Figure 13: Début du code final pour faire fonctionner l'ensemble du prototype

```
while (true){

    capteur_color(&X,&Y,&Z);
        if (X>0.45){            // Permet de trier le rouge
            wait_us(3700000);   // Temps d'attente pour que le cube fait face ua poussoir
            servomot_off();     // Le convoyeur s'arrête face au poussoir
            pousser1();         // Poussoir 1 va pousser le le cube rouge
            tirer1();           // Poussoir 1 va revenir dans sa position initiale
            servomot_on();      // Remet le convoyeur en marche
        poussoir1.period_ms(20); // Bloque les poussoirs dans leurs positions d'origine
        poussoir1.pulsewidth_us(600);
            }
        else if (Y>0.40 && X>0.38){             // Permet de trier le Jaune
            wait_us(8300000);
            servomot_off();
            pousser2();
            tirer2();
            servomot_on();
            poussoir2.period_ms(20);
            poussoir2.pulsewidth_us(600);
        }
        else if (Y>0.40 && X<0.3){              // Permet de trier le Vert
            wait_us(3700000);
            servomot_off();
            pousser1();
            tirer1();
            servomot_on();
            poussoir1.period_ms(20);
            poussoir1.pulsewidth_us(600);

        }
        else if (Z>0.36){                       // Permet de trier le Bleu
            wait_us(8300000);
            servomot_off();
            pousser2();
            tirer2();
            servomot_on();
            poussoir2.period_ms(20);
            poussoir2.pulsewidth_us(600);
            }

  }}
```

Figure 14: Fin du code final pour faire fonctionner l'ensemble du prototype