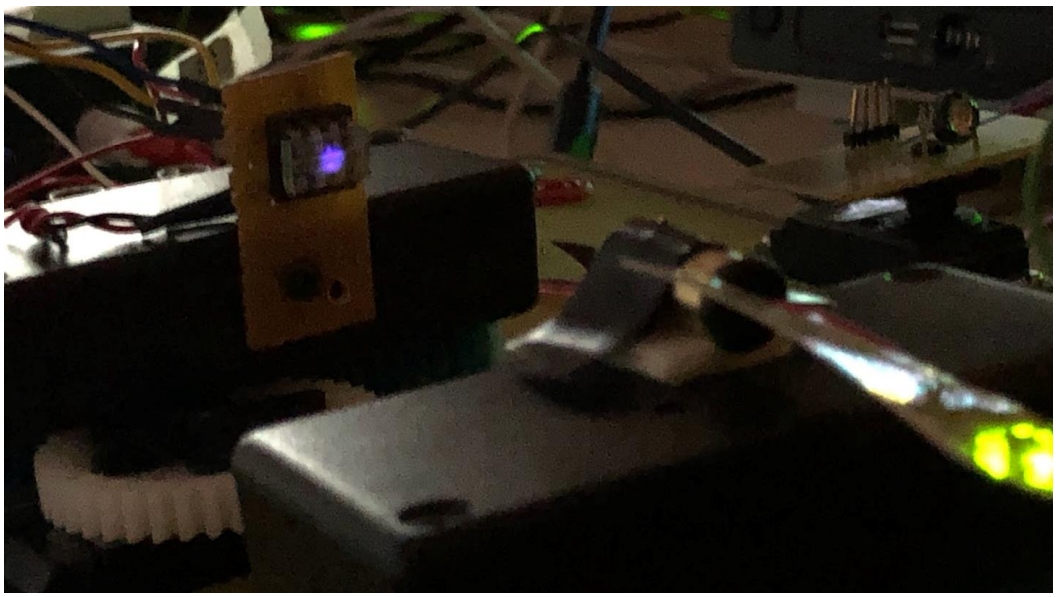
Banc optique pour asservissement de la position d'un laser

Eline Cavadore, Romain Perron, Séléna Rippe, Loïane Thomas

Rapport technique :

Projet PROTIS 2022



1. Présentation générale

1.1 Contexte et présentation du projet

Une entreprise de loisir veut réaliser un parcours laser et a lancé un appel d'offre pour la conception de l'asservissement laser.

Le but est donc de pouvoir suivre le laser avec un CCD qui envoie un signal dès que le laser ne le pointe plus (quelqu'un a traversé le LASER). Le CCD balaye par la suite pour retrouver le laser.

1.2 Cahier des charges

Le système asservi doit être capable de suivre des mouvements de l'ordre de 10 cm/s.

Le laser doit être centré au mieux sur la caméra CCD.

L'interface Matlab doit permettre de modifier le gain du correcteur (gérant ainsi la réactivité de la barrette).

1.3 Contraintes

La motricité de nos composants est gérée par des servomoteurs classiques.

L'asservissement se fera à l'aide d'un correcteur PID numérique.

1.4 Description

Le système est constitué d'une barrette CCD, d'un laser, d'une carte nucléo, d'un servomoteur et d'un ordinateur. Le capteur CCD est asservi en position sur le faisceau laser, et le tout est piloté depuis une interface Matlab depuis l'ordinateur. La carte Nucléo pilote les acquisitions avec le CCD ainsi que son déplacement via le servomoteur, et reçoit des instructions depuis Matlab pour modifier en temps réel le temps d'intégration de la barrette CCD, le gain de l'asservissement, et afficher la distribution d'éclairement dans l'interface Matlab.

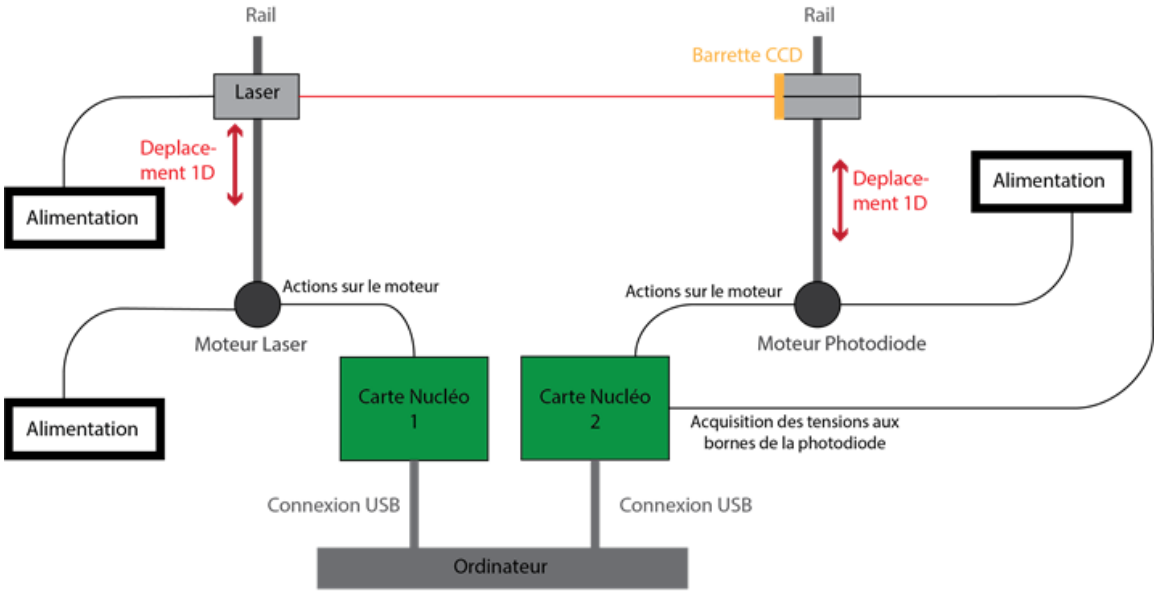


Schéma : Vue d'ensemble du projet

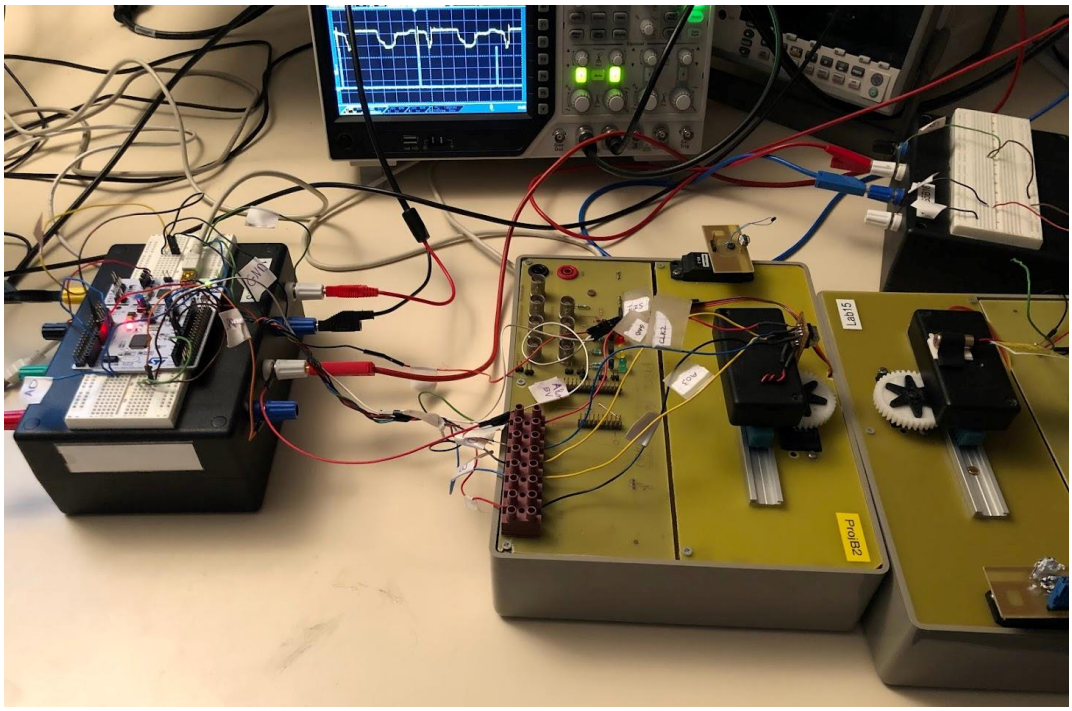


Photo du montage

2. Capteur CCD

Pour fonctionner, la barrette CCD nécessite deux signaux, en plus de l'alimentation 5V :

— Un signal d'horloge CLK, constitué de créneaux de période égale au temps d'intégration Tint souhaité : ce signal permet à chacun des 64 photosites de délivrer un signal de sortie AO. Avec la carte Nucléo, il est généré grâce à une sortie configurée en mode PWM.

— Un signal d'entrée SI, qui passe de l'état logique 0 à l'état 1 pour définir le début d'une séquence de sortie des données. Dans notre cas, l'acquisition se faisant en continu (génération périodique d'un nouveau

signal SI après chaque séquence de sortie des 64 données), les impulsions sur SI sont générées après la 65ème impulsion d'horloge pour que la 64ème valeur puisse être délivrée.

Acquisition CCD à l'oscilloscope : en jaune le signal délivré par la barrette en bleu le signal d'intégration

3. Servomoteur

Un servomoteur est un actionneur qui réalise une rotation d'un angle calibré en fonction d'une commande externe. Cette commande externe est un signal modulé en largeur d'impulsions (MLI ou PWM). Un servomoteur contient un moteur à courant continu, un capteur de position angulaire et un contrôleur numérique. Il se pilote à l'aide de 3 fils : la masse, l'alimentation (ici comprise entre 4,8 et 6V) ainsi que la commande. La commande est un signal rectangulaire numérique de période fixée à 20 ms. C'est la durée du temps haut qui permet de fixer un angle de rotation calibré du servomoteur. Une durée de temps haut de 1,5 ms correspond à une position angulaire de 0°. La plage des angles accessibles par le servomoteur va de -90° à 90°. Les servomoteurs standards atteignent les deux angles extrêmes pour des durées de temps haut de 1 ms et 2 ms. En revanche le servomoteur utilisé dans notre montage ne possède pas ces caractéristiques, il faut donc que le signal de commande ait des temps haut plus longs et plus courts pour atteindre ces angles (3ms).

4. Asservissement

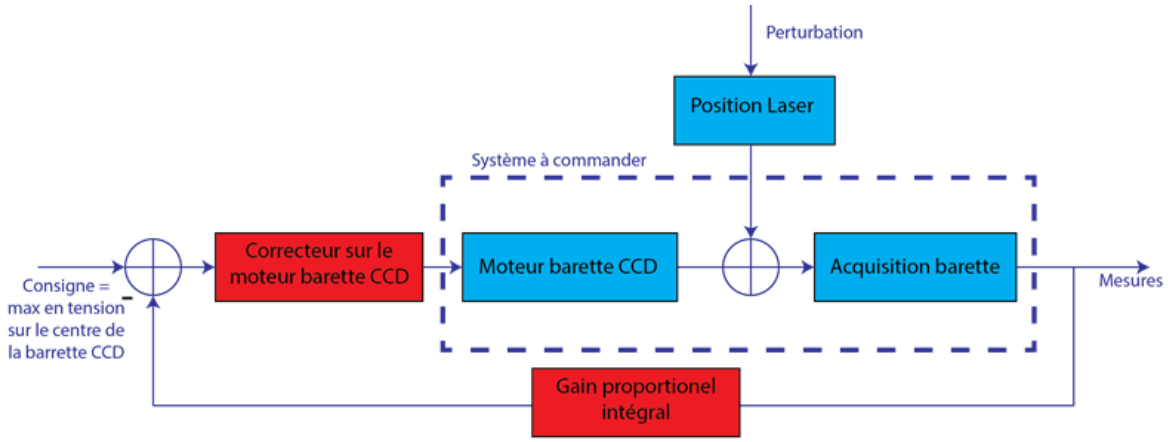
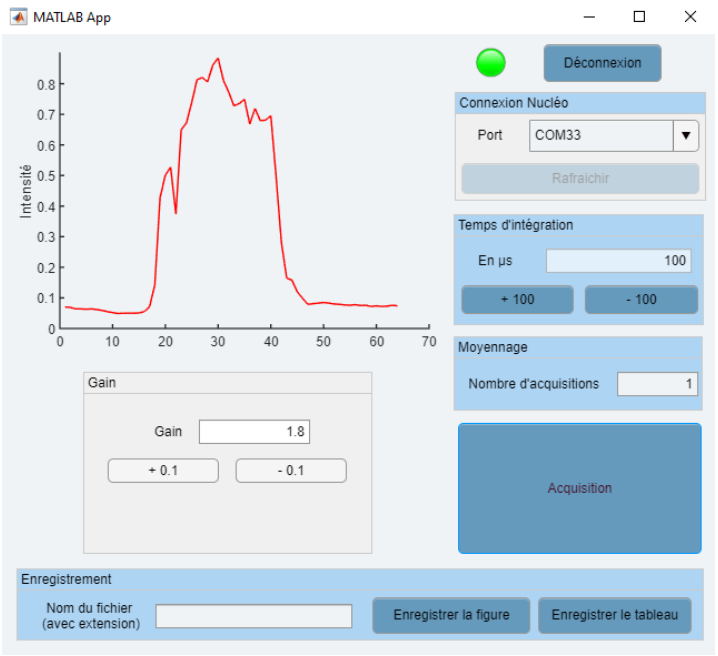


Schéma bloc de l'asservissement

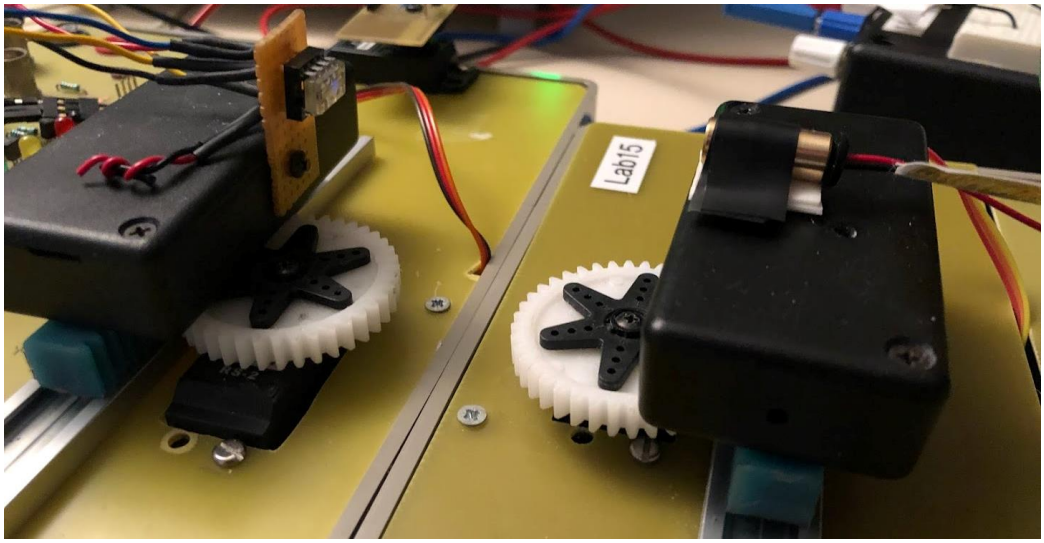
L’asservissement se fait numériquement en utilisant les informations collectées. A chaque temps d’intégration, on relève un vecteur contenant les 64 valeurs d’intensité lumineuse sur chaque pixel. On en cherche le maximum (on prend le dernier pixel maximum s’il y en a plusieurs, ce qui arrive si on sature). Dans le cas où ce maximum dépasse une certaine valeur, on mesure l’écart en pixel entre le centre et ce maximum, que l’on convertit en commande moteur. Un servomoteur prend des valeurs de temps haut comprises entre 1 ms et 2 ms qui fixe sa position angulaire et donc la position du CCD posé sur le support relié au moteur. Nous avons initialement essayé de raisonner sur les angles et sur le rayon du pignon du moteur pour convertir proprement notre écart de pixel en commande moteur. Puis nous avons compris que ces rapports de proportionnalité dépendaient de notre gain, et avons donc fixé des facteurs en produit de notre gain, gain que l’on modifie par une interface Matlab pour avoir le résultat attendu. Cela explique la présence du 1/6 aux lignes 80 et 89 en produit du gain. On pourrait l’enlever du code et diviser le gain par 6 dans notre interface. Une fois la conversion faite, on l’envoie au moteur dont la position angulaire s’adapte pour recentrer le CCD. Une nouvelle mesure est faite, qui donne un nouvel écart plus petit que celui d’avant et une nouvelle position. Le CCD fait donc des aller-retours d’amplitude de plus en plus faible autour du centre jusqu’à ce que le maximum soit compris dans les 5 pixels centraux (l’erreur commise sur la position du laser est donc de 3 pixels).



Interface Matlab utilisée pour modifier le gain

Dans le cas où le maximum ne dépasse pas la valeur de seuil ligne 76 du code, le servomoteur balaye tout le rail. On change les rapports de proportionnalité devant le gain afin que les mouvements dans ce cas-là soit plus ample que quand le CCD pointe le laser.

Augmenter le gain va certes faire balayer le chariot plus vite, mais au détriment de l’amortissement, qui va rendre le pointage plus long voire impossible. Une voie d’amélioration serait donc d’avoir 2 gains, un pour le pointage lorsque le laser est détecté et un pour le balayage du chariot quand le CCD cherche le laser.



Duel entre un CCD et un laser

5. Conclusion et perspectives

Lors de ce projet, nous avons réussi à asservir la position de la barrette CCD pour qu'elle suive le mouvement du laser. Nous avons en particulier répondu aux demandes du cahier des charges.

- Nous avons proposé une solution pour que le spot soit centré au mieux sur la caméra CCD par la recherche du pixel avec la tension maximale. Nous avons tout de même autorisé une incertitude de plus au moins 3 pixels.

- Nous avons mis en place une interface Matlab, simple d'utilisation, pour augmenter ou réduire le gain en temps réel, c'est-à-dire sans devoir modifier le code et le recompiler.

- Nous avons asservi la position de la barrette CCD de telle sorte qu'elle suive un mouvement de translation de vitesse de l'ordre de 10 cm/s. Cette caractéristique a pu être vérifiée en calculant la vitesse de translation du laser à l'aide de la vitesse de rotation du servomoteur donnée par le constructeur à 6 V, et la mesure du rayon de la roue dentée.

Néanmoins notre système possède toujours des limitations.

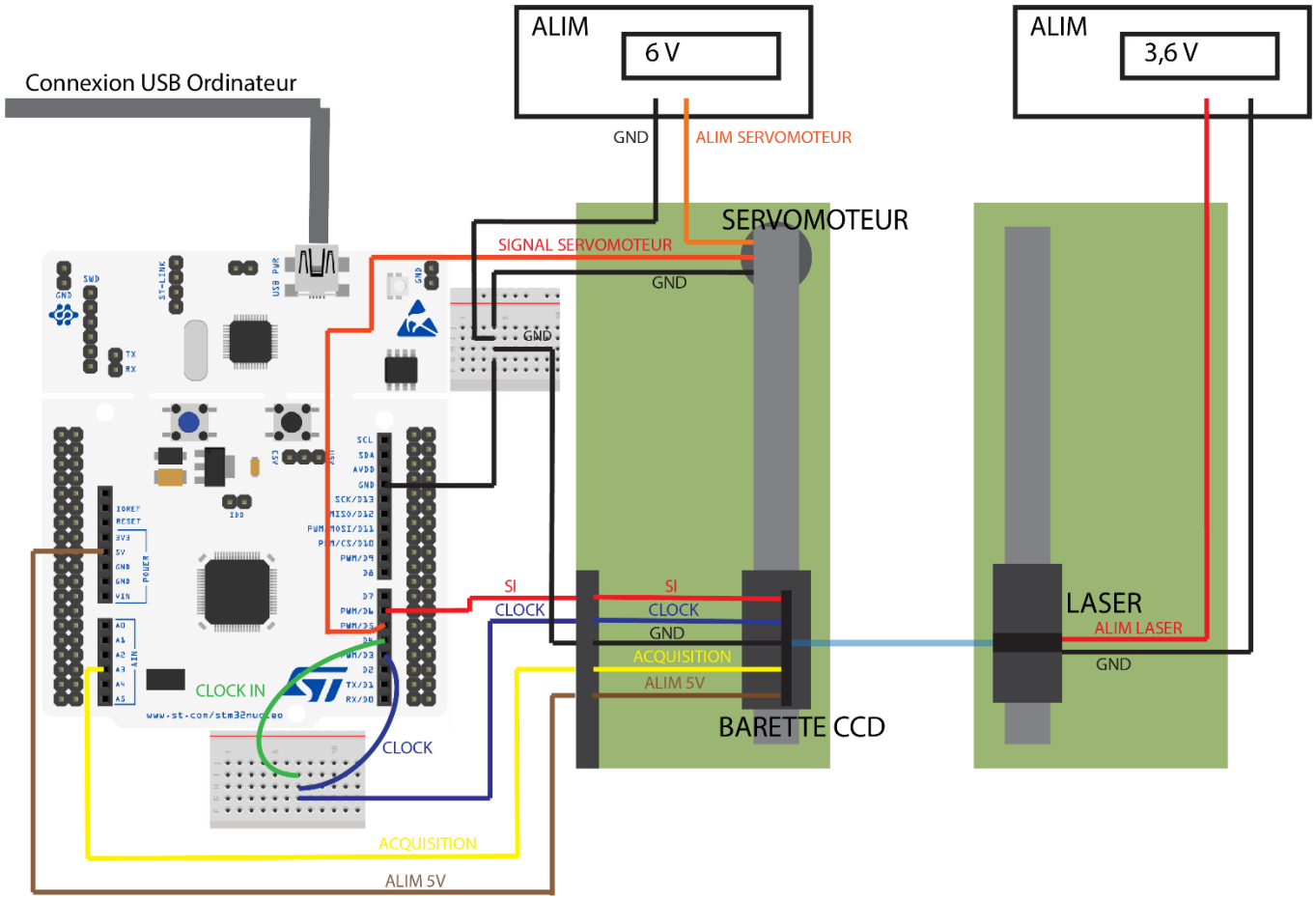
- On remarque que la photodiode oscille lorsque la barrette CCD sature. En effet, la tension maximale aux bornes de la barrette est obtenue pour plusieurs pixels. Notre idée pour remédier à ce problème serait d'améliorer notre critère de centrage sur la barrette. Au lieu de prendre le pixel où le maximum est atteint, on peut prendre le pixel qui correspond au barycentre des intensités.

- Lors de la perte du spot laser sur la barrette CCD, nous aimerions améliorer la recherche du sport laser le long du rail. Par exemple, en estimant le sens vers lequel le laser est parti. Cela peut se faire en regardant le signe du dernier écart en pixel mesuré.

- Notre plus grande limitation a été l'alignement du laser. Nous n'avions pas de support adapté et stable pour notre laser. Par conséquent, lors du mouvement de translation sur le rail, le spot partait vers le haut ou vers le bas à cause de son inclinaison. Ainsi, la plupart du temps, l'asservissement faisait défaut à cause du désalignement du laser.

- Dernièrement, nous avions souvent des faux contacts sur les câbles, en particulier au niveau des câbles d'alimentation du laser.

Annexe 1 : Schéma des branchements



Annexe 2 : Code final

```
#include "mbed.h"

int T_INT = 100; //Temps d'intégration
int a = 64; //Compteur
int tab[64]; //Tableau des valeurs des pixels
int tab2[64]; //Copie de chaque acquisition
char l; //Variable pour obtenir le caractère envoyé depuis Matlab
int indexMax; //Pixel contenant le max de l'eclairement
int Ecart_pixel ; //ecart en pixels entre le centre du CCD et le centre du faisceau laser
double Ecart_T; //conversion de l'ecart en commande pour le moteur (via le gain)
int moteur = 1500; //position du moteur
int b=1;
int dbg=0; //debug
double gain=1.8;  //gain de l'asservissement

PwmOut CLK(D3); //signal d'horloge
InterruptIn CLK_IN(D4); //signal d'horloge rebouclé
DigitalOut SI(D6); //signal d'intégration
PwmOut servo_mot(D5); //servomoteur
Serial pc(USBTX,USBRX); //liaison serie PC
AnalogIn capteur(A0); //sortie du CCD

void acquisition (void); // fonction realisant les acquisitions



int main (){
    pc.baud(115200);
    CLK_IN.fall(&acquisition);//on associe la fonction acquisition à la détection de fronts descendants sur l'horloge
    CLK.period_us(T_INT);//période de l'horloge
    CLK.write (0.5);//rapport cyclique
    servo_mot.period_ms(20); //période du servomoteur
    servo_mot.pulsewidth_us(moteur);
    while (1){

        l = pc.getc();//lettre envoyée depuis Matlab
        if (l == 'a'){
            for (int i =0;i<64;i++) pc.printf("%d\r\n", tab2[i]);//on envoie une acquisition vers Matlab
        }
        else if (l == 'b') {
            T_INT = T_INT +100;
            CLK.period_us(T_INT);//on augmente de 100us le temps d'intégration
        }
        else if (l == 'c') {
            T_INT = T_INT - 100;
            CLK.period_us(T_INT);//on diminue de 100us le temps d'intégration
        }
        else if (l == 'g') {
            gain = gain + 0.1;//on augmente de 0.1 le gain d'asservissement
        }
        else if (l == 'f') {
            gain = gain - 0.1;//on diminue de 0.1 le gain d'asservissement
        }

    }
}

void acquisition (void){
    if (a <64){// tant que l'acquisition n'est pas complète
        double x = capteur.read()*1000.0;
        tab[a]=(int)x; //on collecte les pixels
        a++;
    }
    if(a ==64){ //lorsqu'on a les 64 pixels
        // recherche de l'intensité max
        int max=tab[0] ;
        indexMax=0;
            for (int i =0;i<63;i++) {
               if (tab[i]>=max) {
                   max=tab[i];
                   indexMax=i;
                }
            }


        if((max>300)){// si le max est significatif (pour ne pas asservir sur du bruit)
            Ecart_pixel=indexMax-31 ;//distance du max de l'intensité par rapport au centre de la barette
            b=1 ;
```

```
79              if(abs(Ecart_pixel)>2){// si le faisceau n'est pas au centre (marge de 2 pixels de chaque côté)
80                  Ecart_T=gain*Ecart_pixel/6 ; //calcul du déplacement moteur
81                  moteur=moteur+int(Ecart_T); //actualisation de la valeur du déplacement
82                  servo_mot.pulsewidth_us(moteur);//déplacement du moteur
83
84
85              }
86          }
87          else { // si le CCD perd le faisceau laser, il fait des aller-retour jusqu'à redétecter un maximum significatif
88              if ((moteur<3000)&(b==1)){ //déplacement dans le sens 1
89                  moteur=moteur+int(gain*64/6);
90                  servo_mot.pulsewidth_us(moteur);
91              }
92              if((moteur>=3000)){//déplacement dans le sens -1
93                  moteur=moteur-int(gain*64/6);
94                  servo_mot.pulsewidth_us(moteur);
95                  b=-1;
96              }
97              if ((moteur>300)&(b==-1)){//déplacement dans le sens -1
98                  moteur=moteur-int(gain*64/6);
99                  servo_mot.pulsewidth_us(moteur);
100             }
101             if((moteur<=300)){//déplacement dans le sens 1
102                 moteur=moteur+int(gain*64/6);
103                 servo_mot.pulsewidth_us(moteur);
104                 b=1;
105             }
106         }
107
108         if(dbg==1){//debug
109             pc.printf("Index Max=%d\r\n", indexMax);
110             pc.printf("Max=%d\r\n",max);
111             pc.printf("\r\n");
112             pc.printf("Moteur=%d\r\n", moteur);
113             pc.printf("\r\n");
114             pc.printf("caractère envoyé=%c\r\n", l);
115             pc.printf("\r\n");
116         }
117
118         for (int j =0;j<64 ;j++) tab2[j] = tab[j];//copie d'une acquisition pour envoi vers Matlab
119
120             wait_us(T_INT);
121             SI = 1;// nouvelle acquisition
122             wait_us(T_INT);
123             SI = 0;
124             a = 0;
125     }
126 }
```