



PIMS 2020-2021: Asservissement laser

Etudiants :

Igor Reshetnikov

Théo Martin

Simon Steinlin

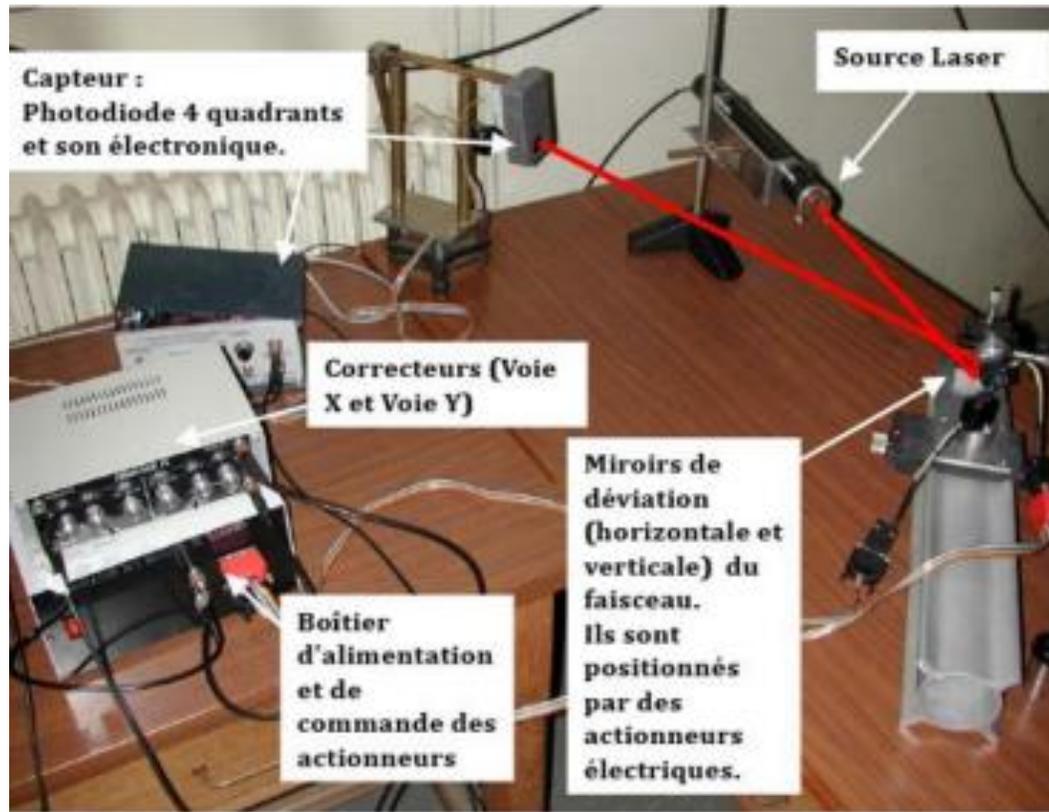
Guillaume Huber

Cyrille Des Cognets

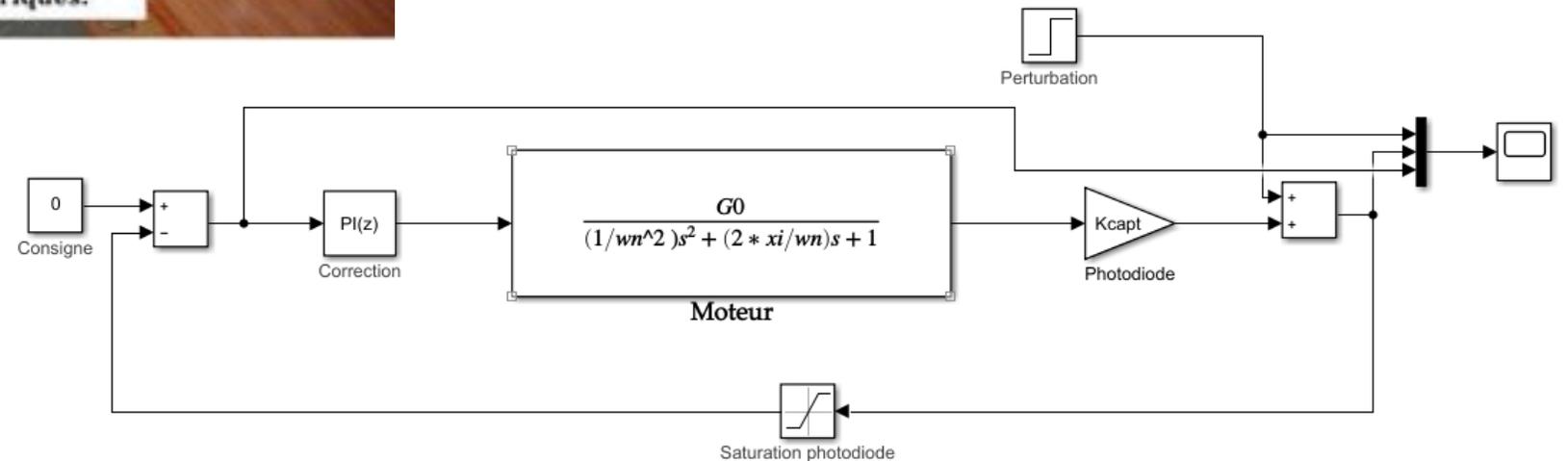
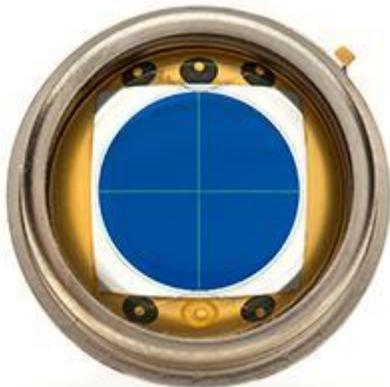
Encadrants :

Julien Villemejeane

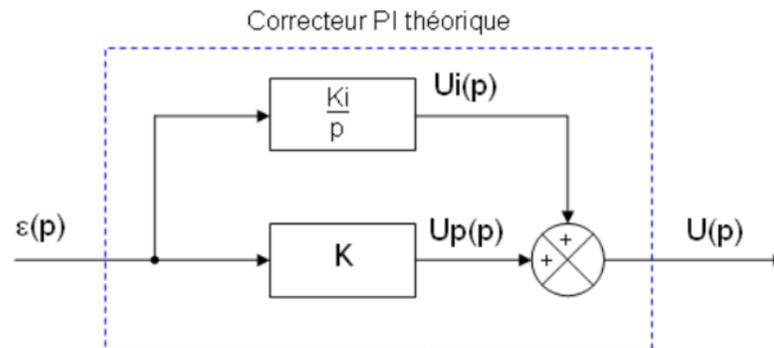
Caroline Kulcsar



Présentation du montage actuel



Correction du système



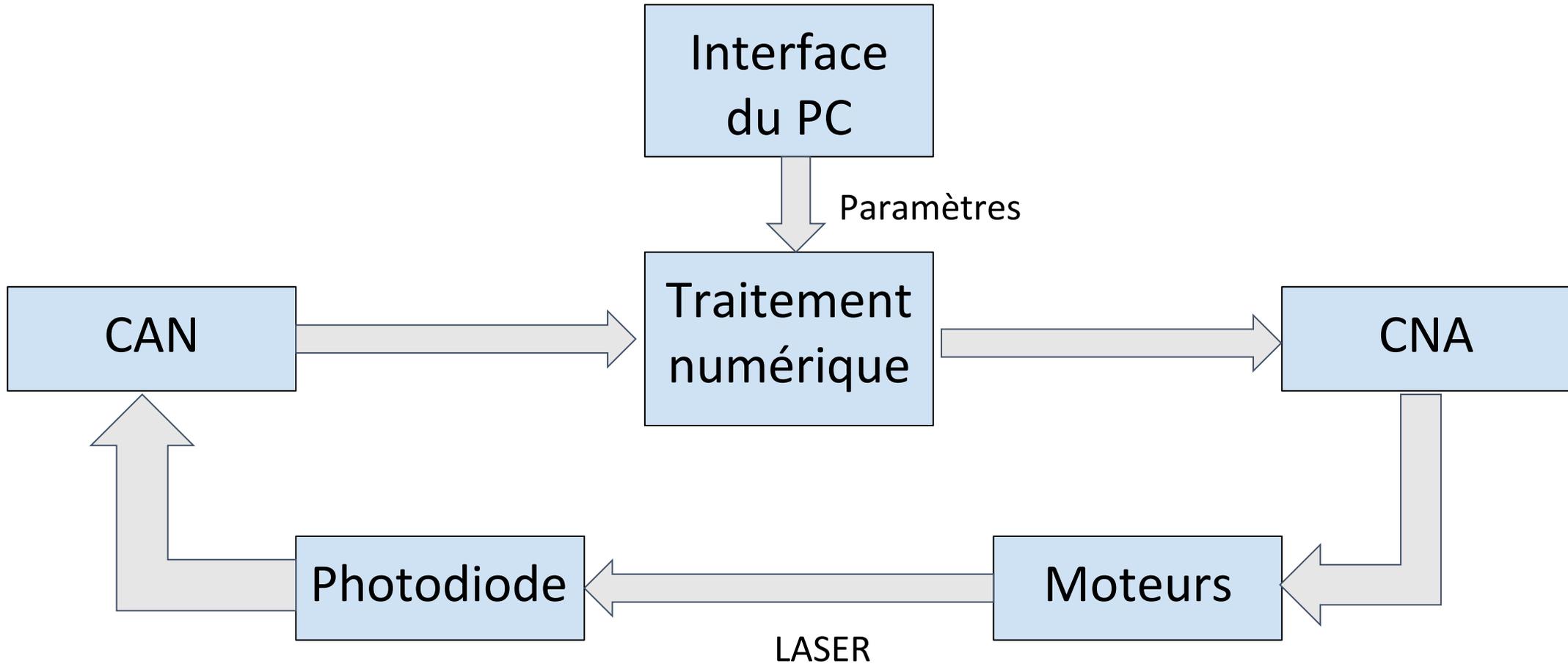
Problématique

- Performance limitée avec un PI
- Paramètres non quantifiés
- Manque de repère sur certains réglages



Correction numérique

Principe de fonctionnement du montage envisagé

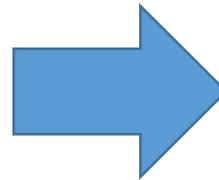


Interface, Choix du langage

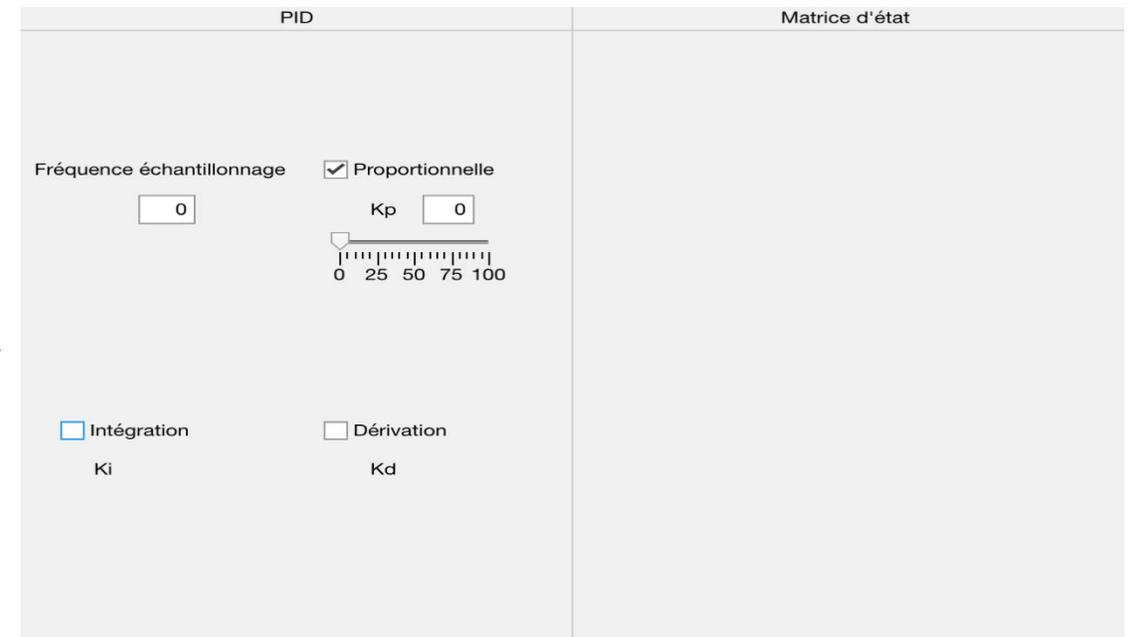
MATLAB	Python
<p>Avantages :</p> <p>Simple dans la création et dans l'utilisation</p> <p>Tous les modules préinstallés</p> <p>Inconvénient : Licence</p>	<p>Avantages : -Universel -Beaucoup de possibilités</p> <p>Inconvénient: Modules à installer et dépendant des versions (communication, interfaçage..)</p>

Présentation (du début) de l'interface

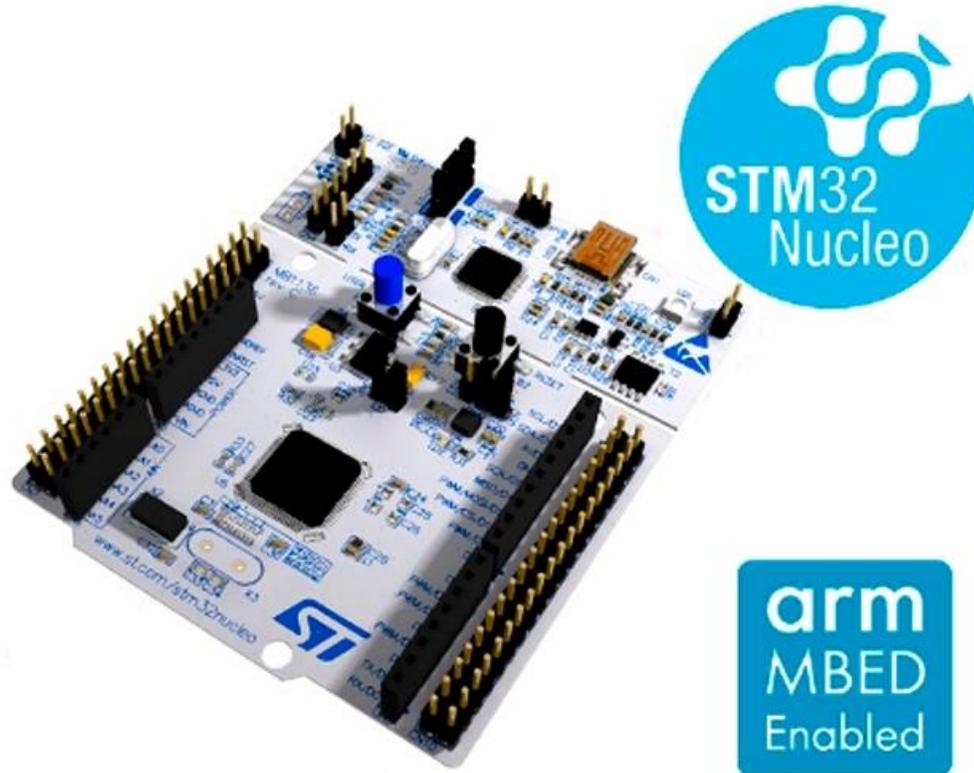
Ancienne « interface »



Nouvelle interface sur ordinateur



Microcontrôleur – carte Nucleo

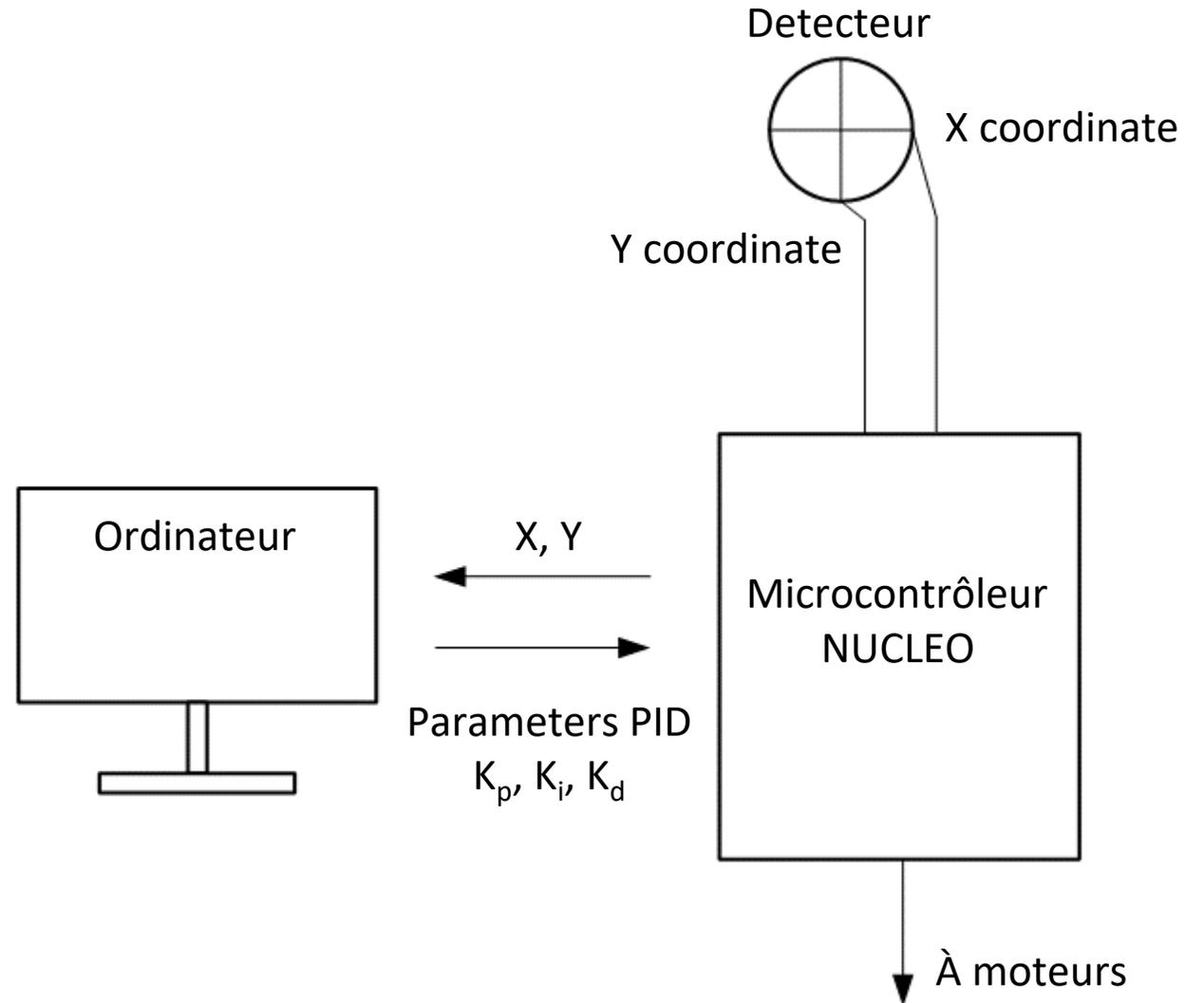


Carte Nucleo-64 STM32L476RG

Protocole de communication

Le modèle de communication le PC et le microcontrôleur :

- Le détecteur mesure les coordonnées et les envoie au contrôleur NUCLEO
- Le contrôleur envoie les données à l'ordinateur
- L'utilisateur peut modifier le coefficient du régulateur PID via PC
- Le contrôleur NUCLEO reçoit les nouveaux coefficients PID et contrôle les moteurs



Code de microcontrôleur

Le modèle de communication le plus simple
entre la carte et le PC :

LED clignotant toutes les 2s

Si un utilisateur appuie sur le bouton,
la carte envoie l'état de la LED au PC

```
10 int main()
11 {
12     Serial pc(USBTX, USBRX);
13     pc.baud(115200);
14     int T_temp=0;
15     int T_glob=0;
16
17     DigitalOut led (LED1);
18     DigitalIn button (USER_BUTTON);
19
20     double LEDFreq=2000; //Period of LED ,ms
21
22     double Button_State_Last=0; // Previous button state: 0 - pushed, 1 - not
23
24     int Value=0; // Value to transmit, LED state
25
26     while (true) {
27         if(T_temp>=LEDFreq) // LED blinking
28         {
29             T_temp=0;
30             led=!led;
31         }
32
33         if(Button_State_Last==1 && button==0)
34         {
35             Value=led;
36             pc.printf("%d\n %d\n %d\n", Value, T_temp, T_glob);
37         }
38
39         Button_State_Last=button;
40         thread_sleep_for(1); //1ms
41         T_glob++;
42         T_temp++;
43     }
44 }
```

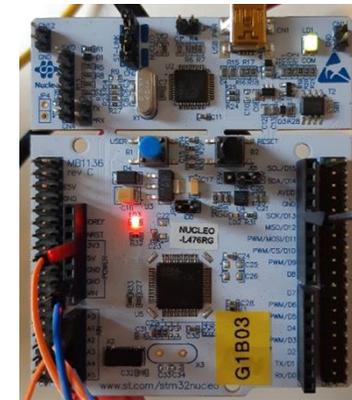
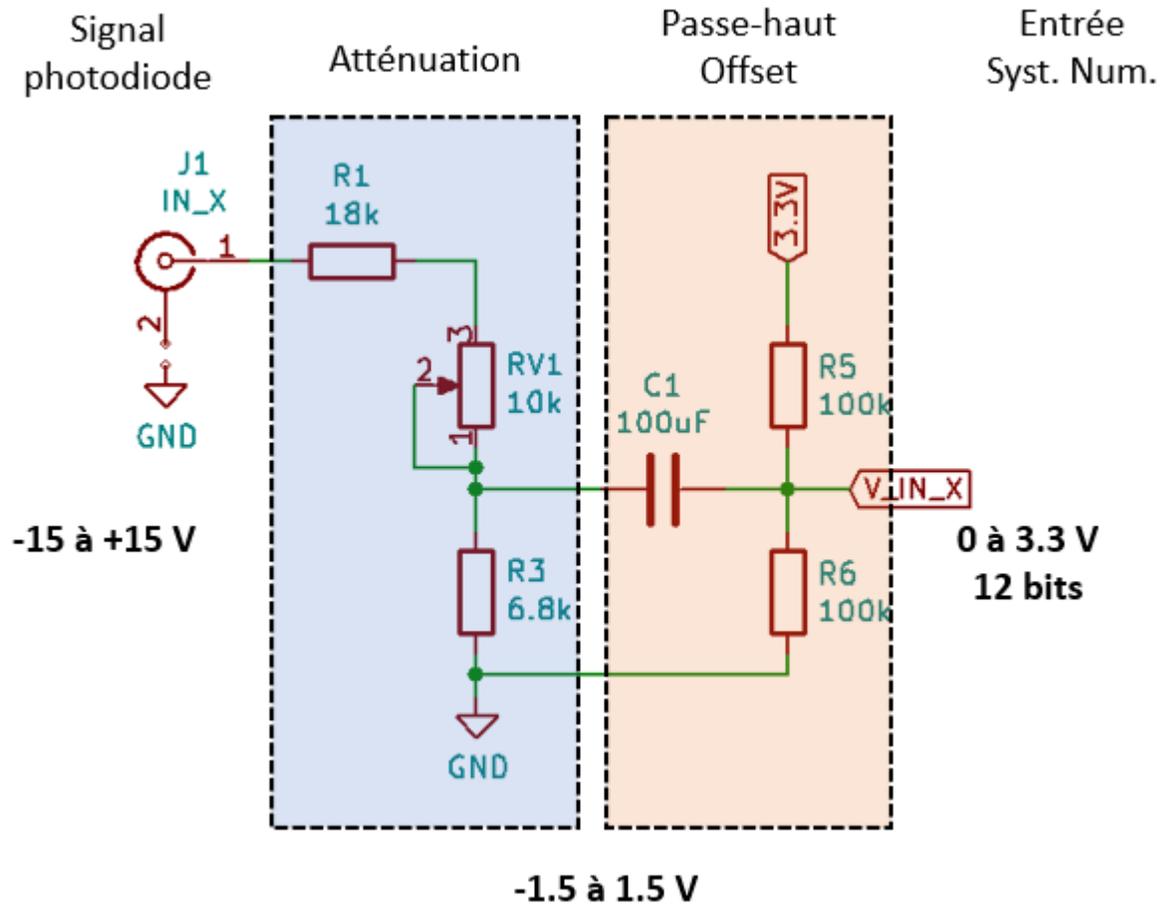
Code de MATLAB

```
1
2  nucleo = serial('COM9', ...
3                'BaudRate', 115200, ...
4                'Parity', 'none', ...
5                'DataBits', 8, ...
6                'StopBits', 1);      %change depending on mbed configuration
7  fopen(nucleo)
8
9  while (true)
10 Nucleo_Output=[0,0,0];
11 for k=1:3
12     Nucleo_Output(k)=fscanf(nucleo, "%d");
13 end
14 Nucleo_Output
15 end
16
17
18 fclose(nucleo);
```

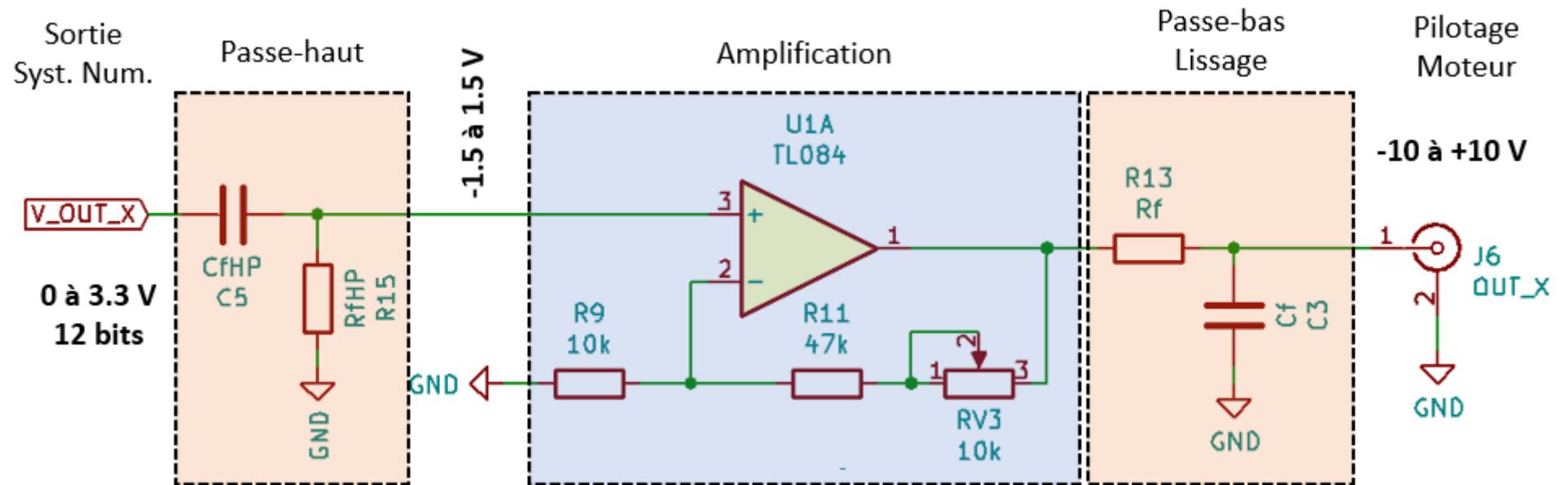
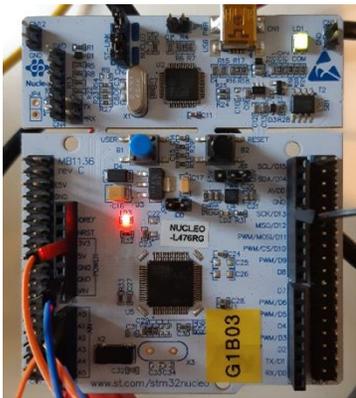
Nucleo_Output = 1x3			
1	86	136336	
Nucleo_Output = 1x3			
1	154	138904	
Nucleo_Output = 1x3			
0	102	140602	
Nucleo_Output = 1x3			
1	243	141993	
Nucleo_Output = 1x3			
1	69	143319	
Nucleo_Output = 1x3			
1	47	144797	

Le code MATLAB pour recevoir et lire les données envoyées par carte

Adaptation de tension



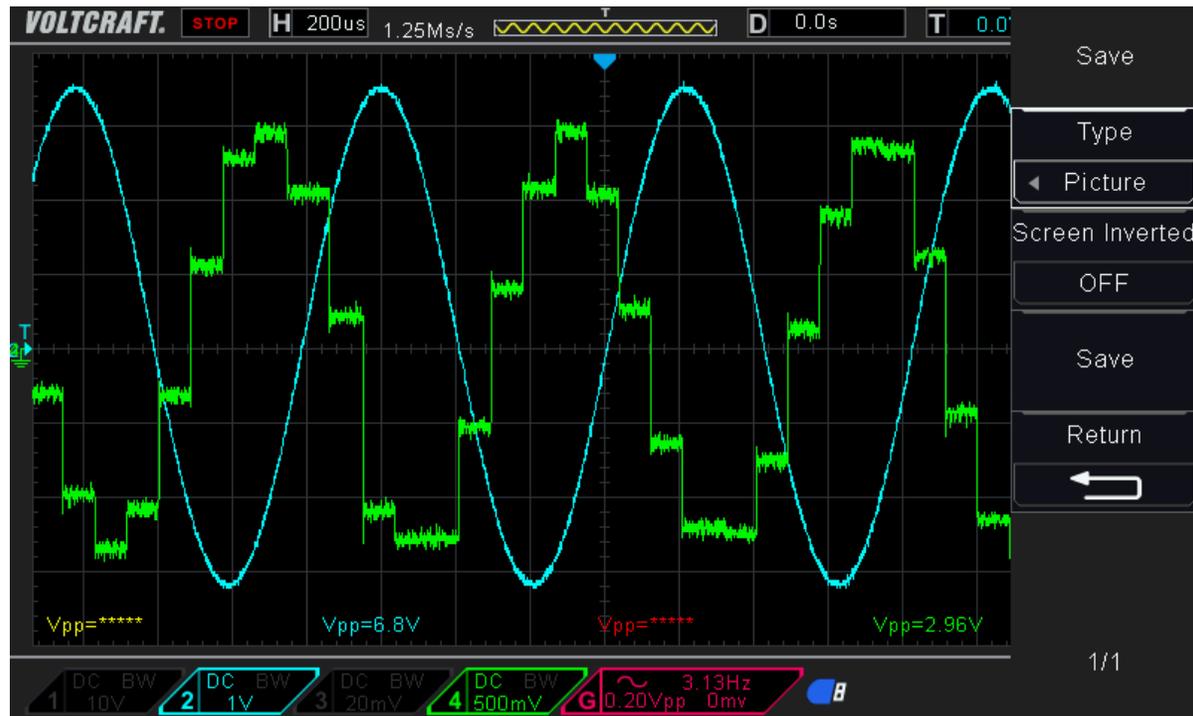
Adaptation de tension



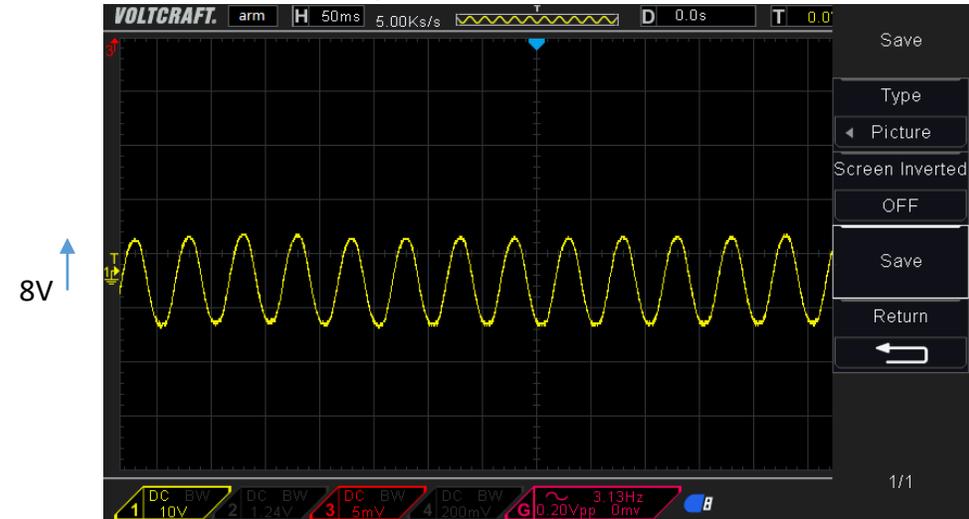
Code nucléo

```
33 int main(){
34     //Le tableau qui contiendra les valeurs précédentes
35     for (k=0;k<Ti;k++){
36         TX[k]=0.5;
37     }
38
39     for (k=0;k<Ti;k++){
40         TY[k]=0.5;
41     }
42     while(1){
43
44         while (i<Ti){
45             i+=1;
46             measX=InputX.read();
47             measY=InputY.read();
48             erreurX=measX-0.5;           // On a l'erreur par rapport à 0
49             erreurY=measY-0.5;
50             proportionnelX=K_p*erreurX; //Partie Proportionel
51             proportionnelY=K_p*erreurY; //Pour un K=1, le correcteur ne fait rien et pour K=-1 est inverseur
52             TX[i]=erreurX;             //Partie intégrateur
53             TY[i]=erreurY;
54             integrateurX=sum(TX,Ti)/(Ti*(33e-6)); //Sum/(Ti*periode d'échantillonnage)
55             integrateurY=sum(TY,Ti)/(Ti*(33e-6));
56             OutputX.write(0.5+proportionnelX); //Output
57             OutputY.write(0.5+proportionnelY);
58         }
59         i=0;
60     }
61 }
```

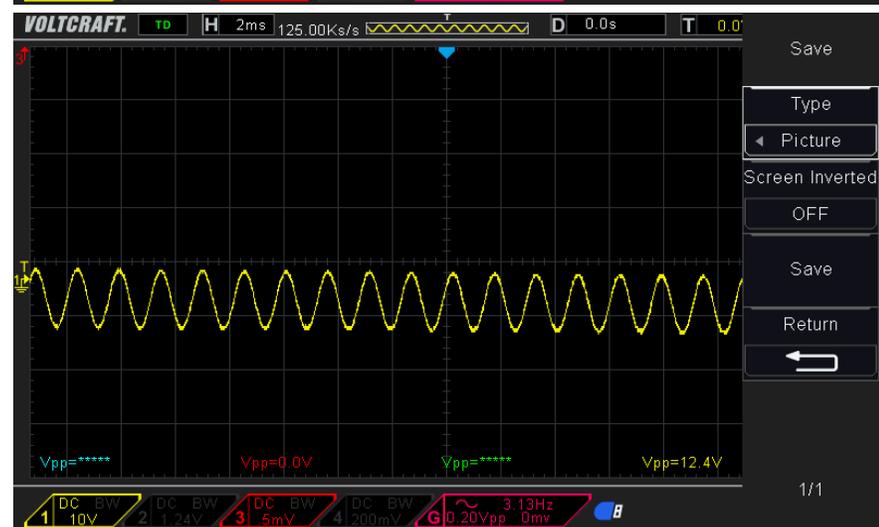
Fréquences caractéristiques et Gain



Pour une amplitude d'entrée de 70mV

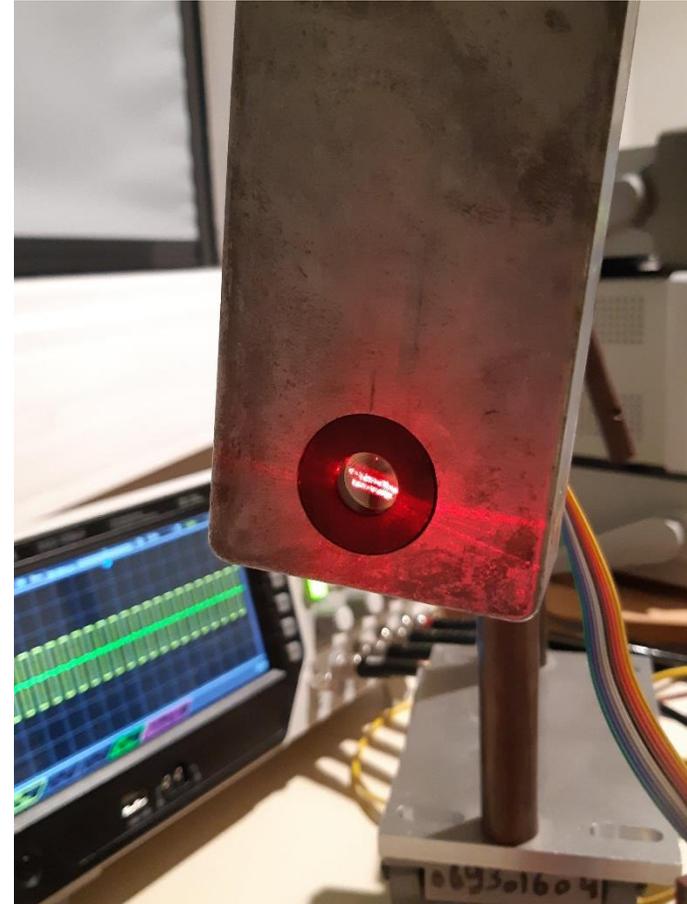
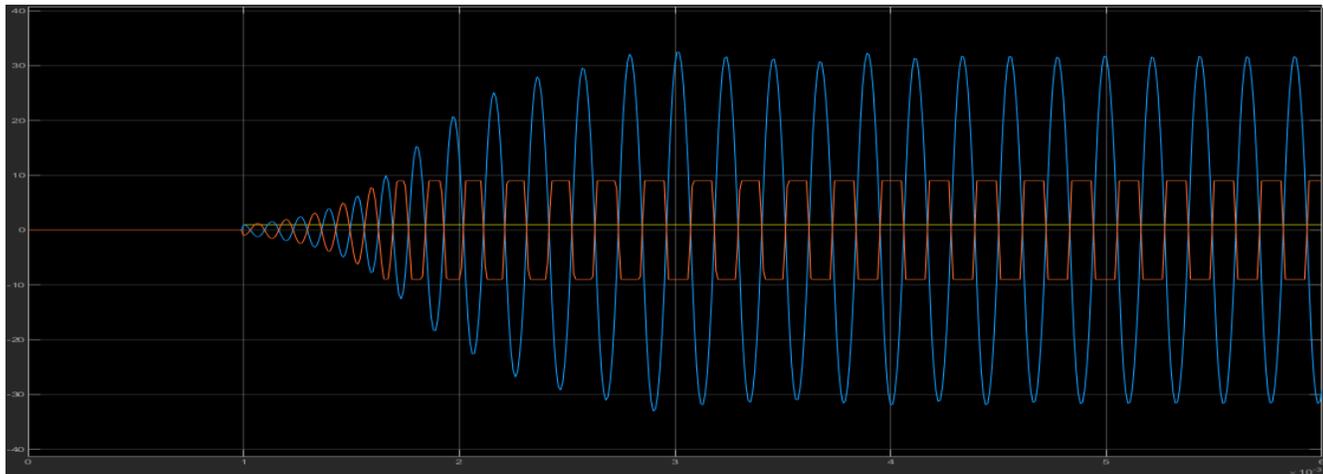
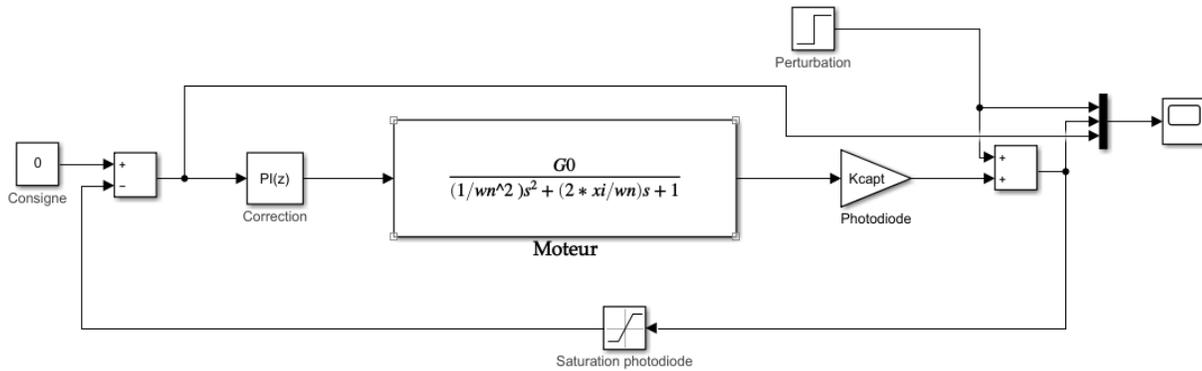


Tension en sortie de la photodiode en appliquant une tension sur les miroirs au cours du temps

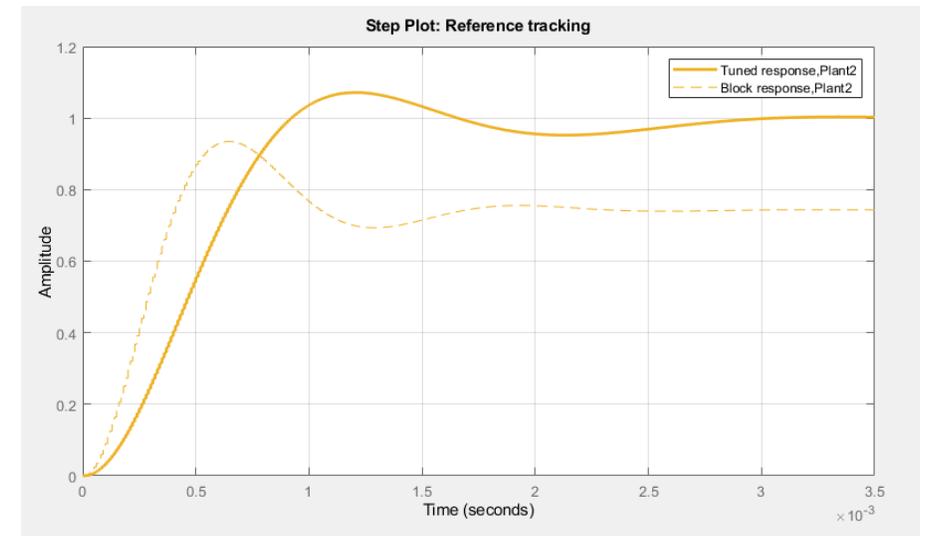
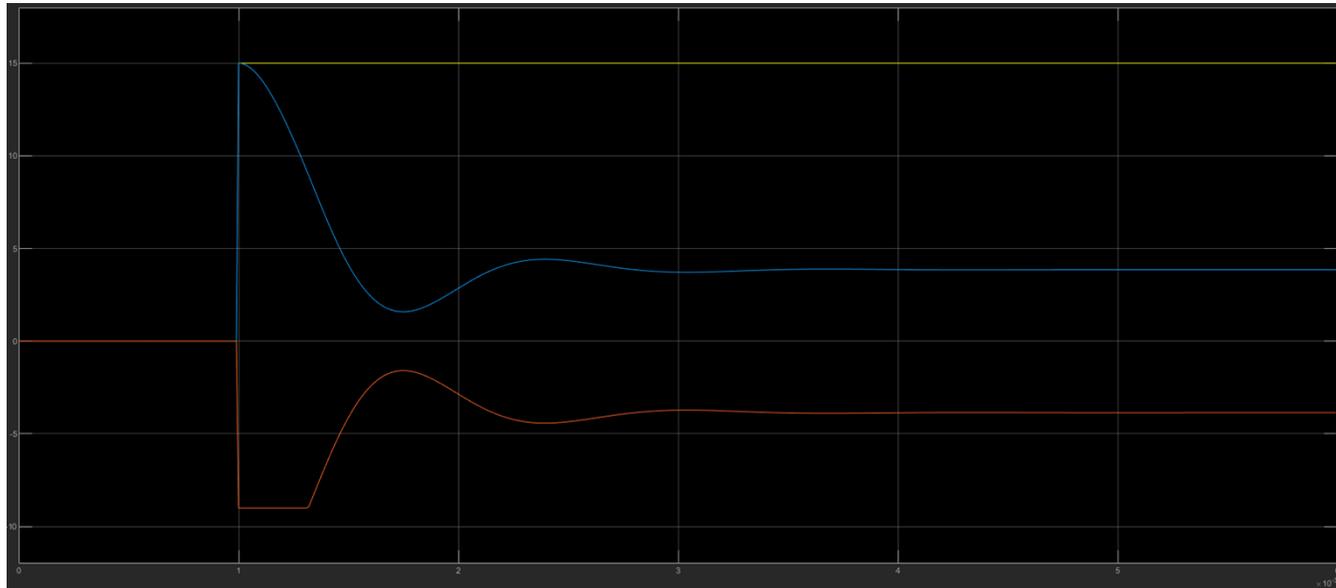


F_c=650Hz

Simulation sur Matlab

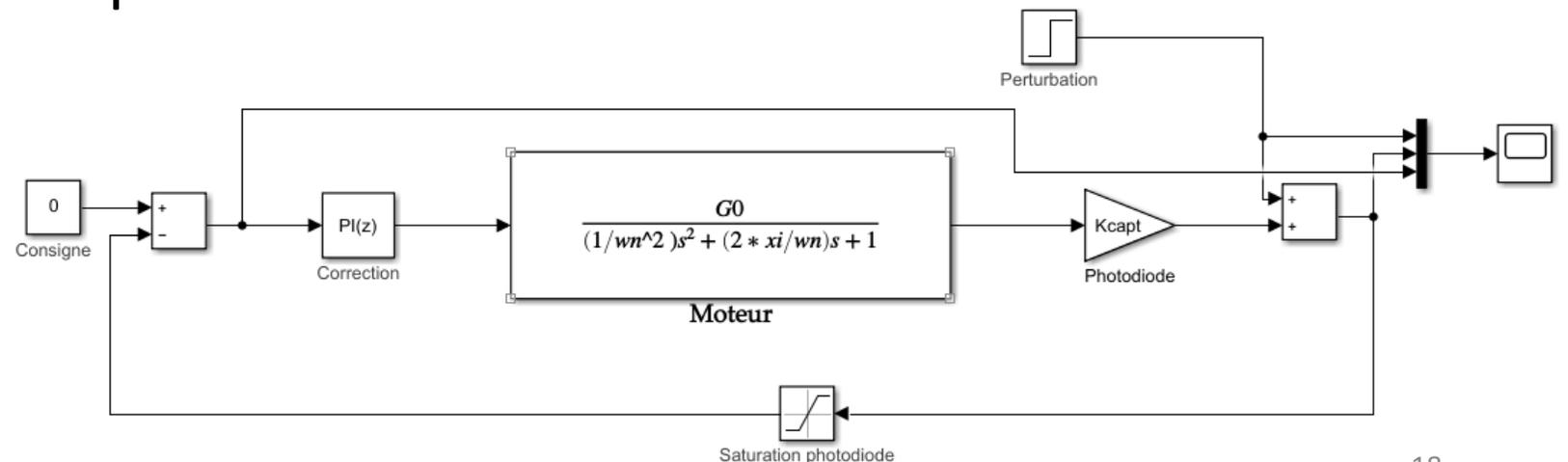
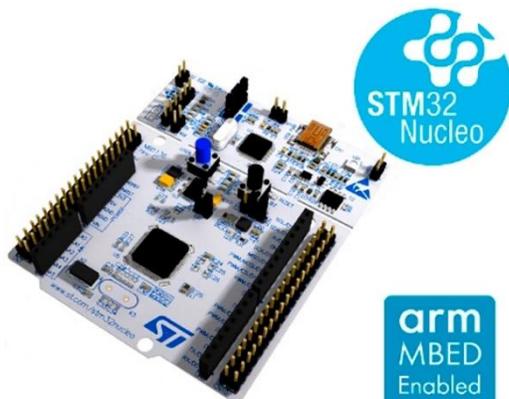
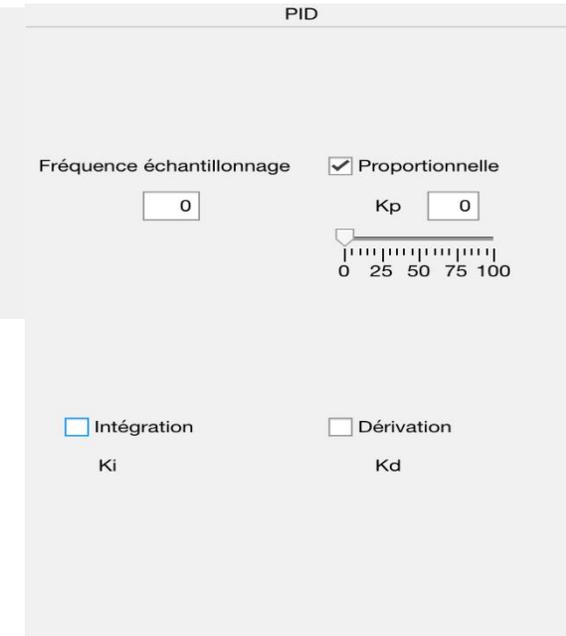
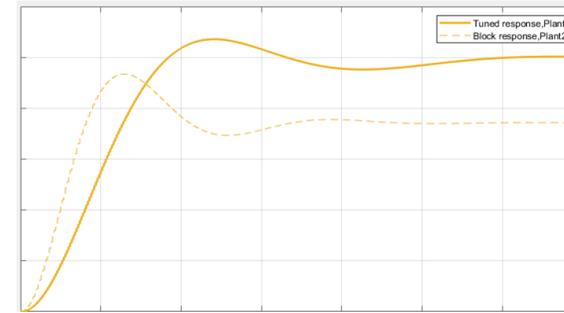


Correction PI



Bilan de la semaine

- Prise en main de la manip
- Acquisition de connaissances en automatique
- Prise en main Mbed pour piloter une carte Nucleo
- Développement d'une interface de pilotage du système
- Modélisation de la manip sous Matlab



Problèmes à traiter

- Mise en œuvre de la correction numérique (PI/D)
- Choix des moteurs pour une application TP
- Réalisation d'une carte électronique
- Utiliser une théorie prédictive

Sources

- Les étoiles laser artificielles (Costel SUBRAN)
- Thèse sur l'optique adaptative de Lucie LEBoulleux
- Datasheet des moteurs scaners