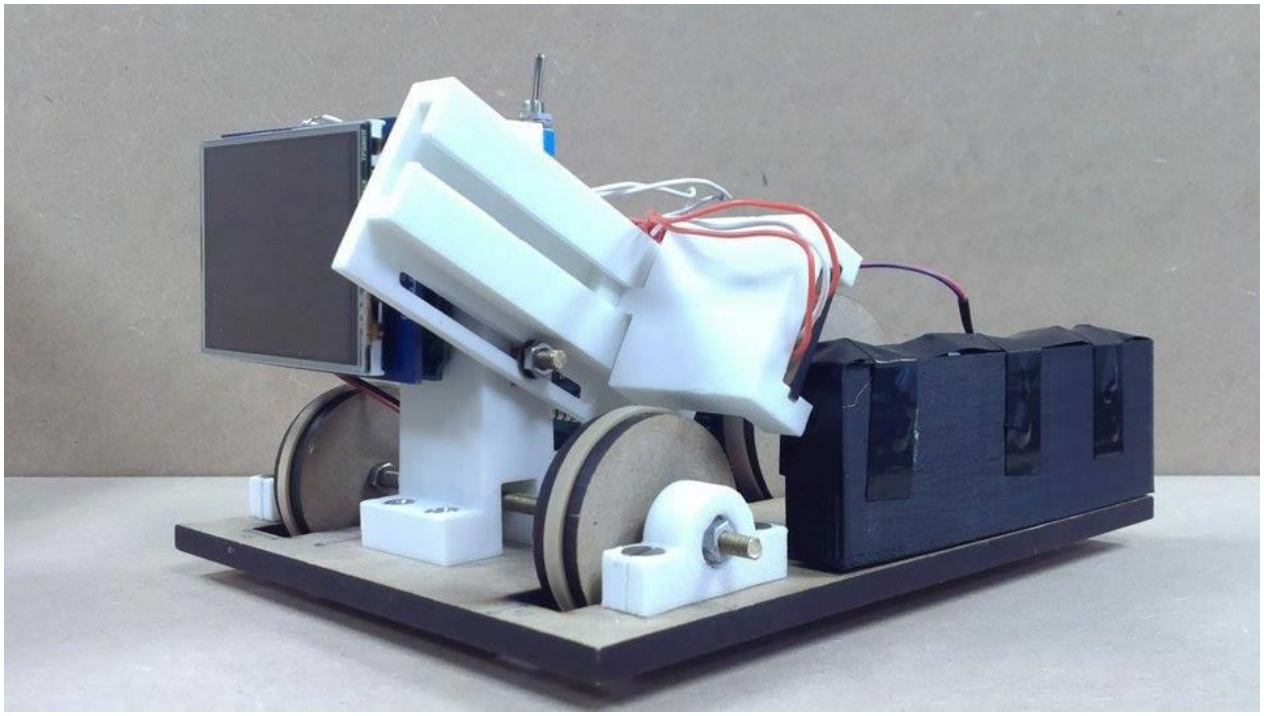
PROJET ARE 2016 :

PROFILOMETRE



Réalisé par : BERRABAH Malissa, DEPALLE Théo, ACTUARIUS Tristan, TEIAR Samy, CHICHE Simon, SAMY Patrick et CECCARELLI Julien.

Sommaire :

I- Introduction :

Dans le cadre de notre ARE, les professeurs nous ont demandé de réaliser un profilomètre.

Qu'est-ce qu'un profilomètre ?

C'est un instrument utilisé pour mesurer le relief d'une surface, notamment dans le but d'en évaluer la rugosité ou la micro-géométrie. C'est en réalité une sorte de scanner 3D, mais en 2D.

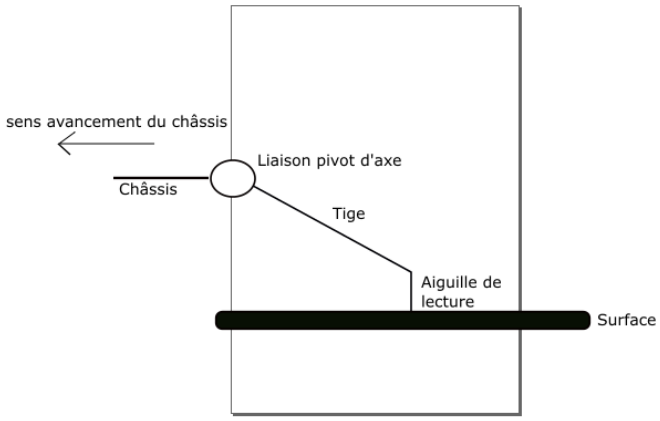
Les profilomètres sont à leur origine dotés d'une pointe très fine en diamant qui lit l'altitude lorsqu'on la déplace le long de la surface. Ce principe est toujours très utilisé, mais il est aujourd'hui complété par nombre de dispositifs optiques.

Le but de notre projet est de réaliser un profilomètre capable de détecter une variation de surface de l'ordre de quelques micromètres.

Suite à un travail de recherche de plusieurs heures, nous trouvons qu'il existe différents profilomètres, les principaux étant les profilomètres optiques et les profilomètres à contact, mais nous ne savons pas vraiment lequel choisir, et ne trouvons pas énormément de support internet et de DIY pour nous aider. Nous avons juste décidé que le profilomètre se présentera sous la forme d'une "voiture" c'est à dire d'un châssis sur roue, qui avancera à très faible vitesse pour avoir le temps de détecter correctement les aspérités de la surface. Nous aurons donc 4 grandes parties dans notre projet : motrice, détection, analyse et conception 3D. Mais nous nous rendons vite compte que ce sera un projet difficile à réaliser et apercevons de nombreuses difficultés, notamment pour pouvoir capter de très faibles variations de surface de l'ordre d'une dizaine de micromètre.

La détection de la variation de la hauteur d'une surface étant le point central de notre projet, nous allons y consacrer une grosse partie de notre temps. Pour commencer nous décidons que pour la détection nous utiliserons une sorte de “tête de lecture” qui sera en contact avec la surface.

Tout d'abord nous voulons que la tête de lecture se présente comme une tige accrochée au châssis au bout de laquelle il y aura une aiguille en contact avec la surface, qui va lire ses variations.



<u>Lecture des aspérités</u> :

Nous nous sommes attardés sur trois solutions pour la lecture des aspérités.

1. <u>Tête de lecture de disque vinyle</u> :

<u>Avantages</u> :	**<u>Inconvénients :</u>**
- Détecter des variations de quelques dizaines de microns. - Capteur déjà fait. - Tête de lecture très résistante.	- Travail de décomposition des signaux. - Tête de lecture qui risque de rayer certaines surfaces. - Prix élevé.

2. <u>Tige piézoélectrique</u> : *qui change de résistance sous l'action d'une contrainte mécanique.*

<u>Avantages</u> :	**<u>Inconvénients :</u>**
- Capteur peu coûteux. - Signal facile à analyser.	- Flexion très faible et donc une variation de résistance trop faible pour être détectée.

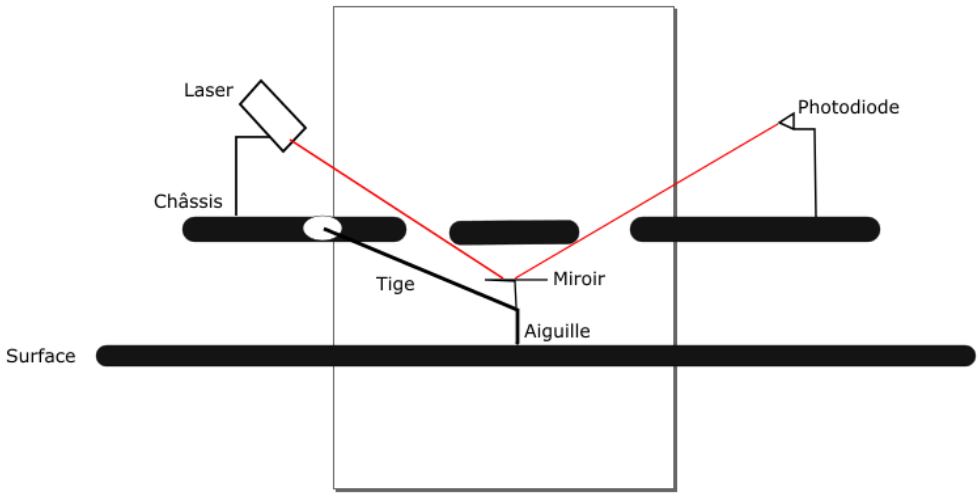
3. Détection optique : *nous développerons en particulier cette solution puisque c'est celle que nous avons choisi.*

Comment ça marche ?

Sur le papier ça a l'air simple, nous restons sur une tige attachée par l'extrémité à un châssis par une liaison pivot d'axe, et une aiguille de lecture de l'autre côté. Mais sur cette aiguille nous rajoutons un miroir posé horizontalement.

Nous attachons au châssis : un laser et une photodiode. Le laser pointera vers le miroir avec un certain angle, le faisceau sera alors réfléchi vers la photodiode qui captera l'intensité lumineuse de celui-ci. Quand l'aiguille de lecture changera de hauteur, l'angle incident du faisceau sera alors modifié et la photodiode ne captera plus la même intensité lumineuse. Nous pensons que c'est la meilleure solution pour détecter de très faibles variations de l'ordre du micron tel que nous le voulons.

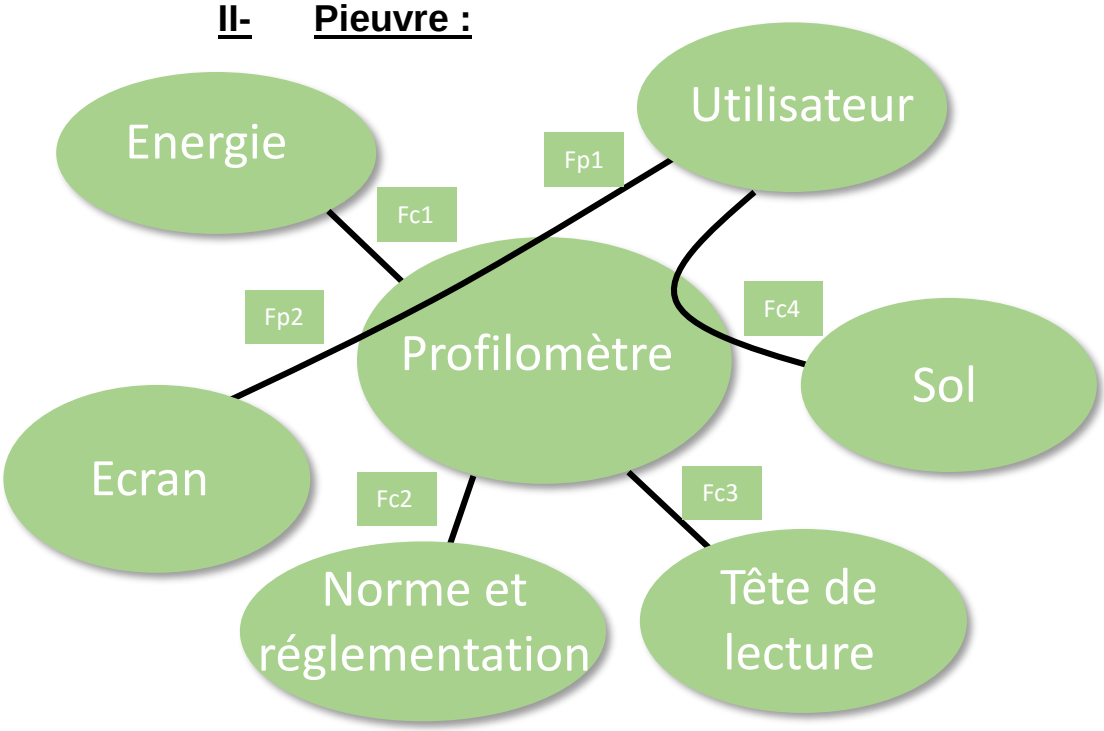
Il y aura néanmoins une grosse partie de calculs, et bien sûr ces calculs ne représenteront que la théorie, il faudra ensuite faire toute une série de tests pour les vérifier, car il nous faut une très grande précision sur l'analyse du signal.



Afin d'avoir des résultats concrets, nous avons effectué un test de détection laser et vérifier qu'une photorésistance (moins précise qu'une photodiode) pourra détecter des variations de luminosité et un test pour étudier les variations de tension en fonction de l'intensité lumineuse d'un laser.

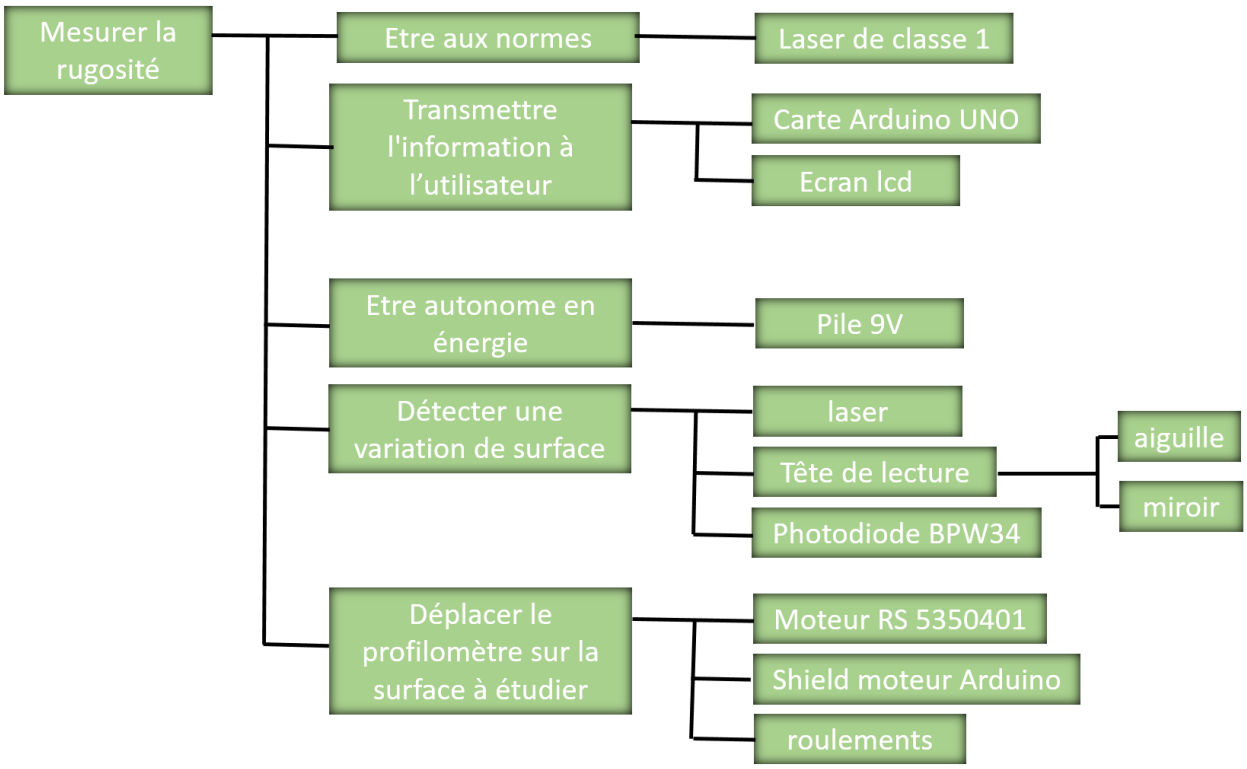
Suite à ces tests, nous pouvons voir que la photorésistance capte de très faibles variations, mais qui sont quand même visibles. Nous pensons donc que notre système de détection fonctionnera.

II- Pieuvre :



Fp1	**Mesurer la rugosité d'une surface**
Fp2	**Transmettre l'information mesurée**
Fc1	**Déplacer le profilomètre sur la surface à étudier**
Fc2	**Etre autonome en énergie**
Fc3	**Détecter une variation de surface**
Fc4	**Etre aux normes**

III- F.A.S.T :



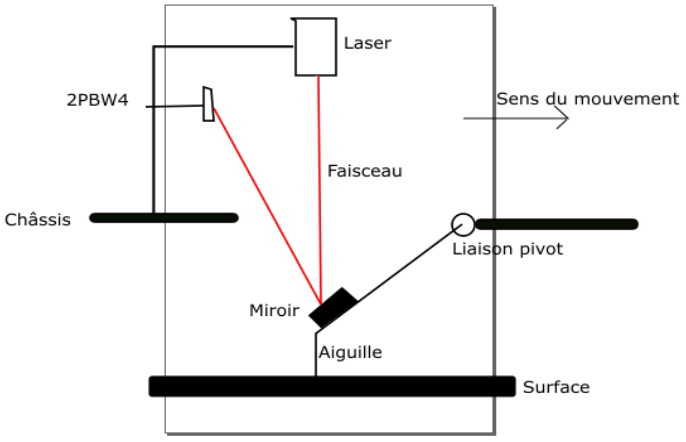
IV- Les photodiodes :

Maintenant que nous savons que nous allons détecter les aspérités avec notre système laser/photodiode, il faut étudier le système (calculs mathématiques). D'une part cela nous permettra de le réaliser, mais d'autre part cela nous permettra de connaître les dimensions de notre profilomètre, pour pouvoir débuter la conception 3D.

Nous avons choisi d'utiliser un laser rouge, puisque c'est le moins cher et le plus courant, de plus c'est celui qui dispose de la longueur d'onde la plus élevée ce qui est le mieux pour notre photodiode qui a une plus grande précision dans des longueurs d'onde autour de 800nm. De plus, nous avons choisis d'utiliser une BPW34, étant une référence dans le domaine de la photo-détection.

Nous décidons de mettre deux photodiodes (BPW34) côte à côte. Au repos, le laser pointera entre les 2 photodiodes qui capteront donc normalement le même signal et après soustraction des signaux sur la carte nous obtiendrons zéro. Quand la pointe de détection bougera, l'angle incident changera et le laser éclairera donc plus une photodiode qu'une autre, en soustrayant le signal on pourra faire la relation entre le signal obtenue et le relief de notre surface

Voici donc à quoi ressemblera notre système de détection final :

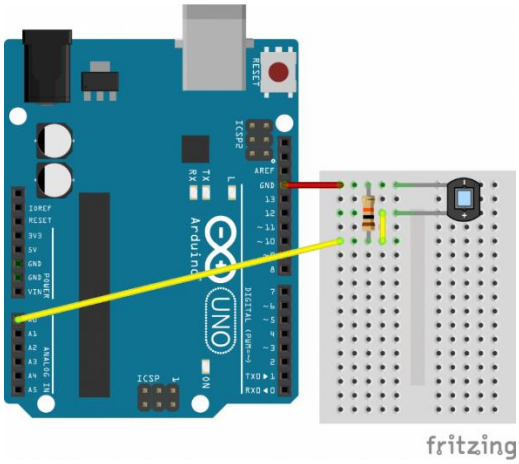


Il faut maintenant voir comment brancher ces 2 photodiodes. Tout d'abord il est important de savoir qu'une photodiode crée **un courant I (en A)** traversée par des radiations

lumineuses. Or notre carte ne détecte que des variations de **tension (en V).** La solution ? **Une résistance** !

En effet **U=R*I** comme l'intensité va changer en fonction de la luminosité, la tension va changer aussi, il suffira donc de brancher la résistance à une entrée analogique et de lire les variations de tension.

Branchements :



Détermination de la distance miroir/photodiode :

Comme il faut que le projet avance, il nous faut un châssis et ses dimensions. Nous avons tout d'abord fait des calculs en nous appuyant sur de la trigonométrie basique. Mais nous nous sommes rendu compte qu'ils étaient en parti faux puisque nous ne tenions pas compte d'un paramètre : le laser ne tape pas toujours au milieu du miroir. Après avoir réalisé ceci nous avons pensé que le calcul était infaisable. Puis après de longues réflexions, nous somme parvenu à cette formule :

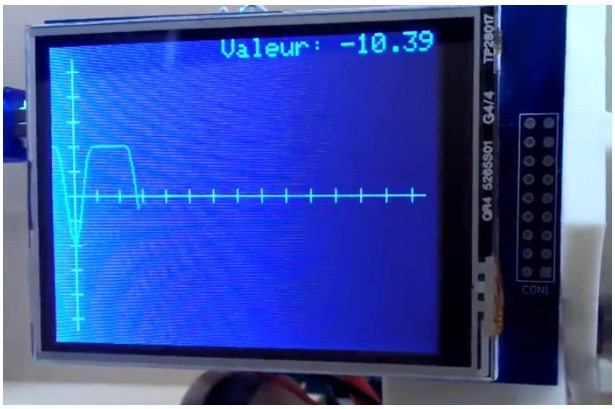
$$l = \frac{\Delta x - \cos(\Delta\alpha)\,\mathrm{si\,n}(2\alpha 2)\left(\frac{\tan(\alpha 2)\cos(\alpha 1)\,2L}{3} - \frac{2H}{3}\right)}{\mathrm{ta\,n}(\Delta\alpha)}$$

- l = longueur miroir/photodiode au point "zéro"
- Δx = la variation maximal de notre faisceau sur les photodiode
- $\Delta\alpha$ = la variation maximum d'angle de la tige
- α_2 = l'angle entre le châssis et la tige au point le plus bas
- α_1 = l'angle entre le châssis et le tige au point "zéro"
- L la longueur de la tige
- H la hauteur entre le haut de l'aiguille et le châssis

Nous voulons une variation *Δx,* d'un maximum de 1mm sur nos photodiodes. Nous avons alors décidé de faire un programme sur Python pour tester différentes valeurs de H et de L pour essayer de trouver la forme la plus "optimisée" : la plus compacte. Et nous trouvons donc une hauteur H de 1cm, un longueur L de 2cm, ce qui donne une longueur l miroir, photodiode de 15.9cm.

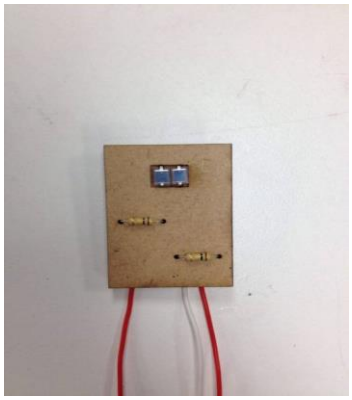
V- Acquisition et affichage :

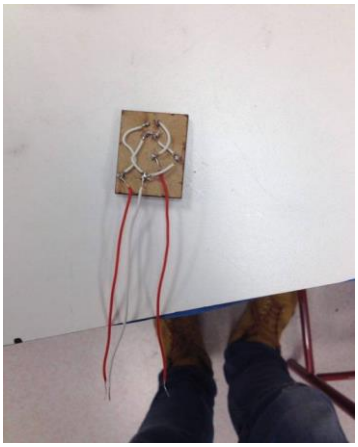
En ce qui concerne l'affichage, nous avons décidé d'afficher notre courbe sur un écran LCD : Ecran TFT 240x340. Il faut tout d'abord savoir que pour faire afficher du texte par exemple à un écran LCD, il vaut mieux trouver une librairie adéquate si on ne veut pas avoir à afficher pixel par pixel. Comme c'est une copie d'un écran d'adafruit, nous avons mis du temps à en trouver une qui fonctionnait. Nous avons programmé l'écran pour qu'il affiche d'abord une "page présentation", puis des axes gradués et une courbe. Puis nous avons enfin branché une photodiode à l'entrée analogique A5 et l'écran a bien tracé la courbe.
Il faut savoir que nous ne pourrons pas tout lier. En effet, le Shield moteur + l'écran LCD + les photodiodes, ça fait trop pour une carte UNO, il faudrait alors une carte Arduino MEGA.



VI- Plaque de détection :

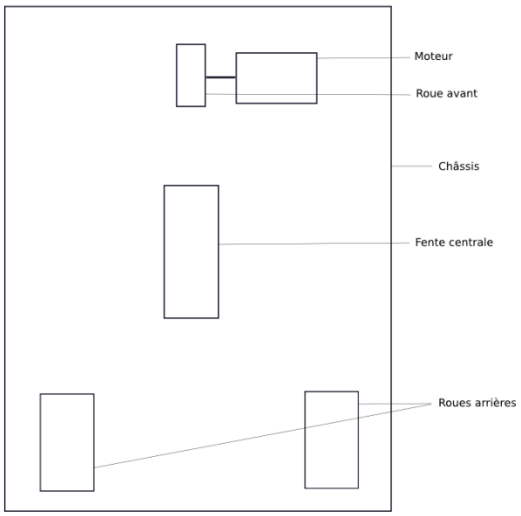
Comme nous ne sommes pas parvenus à faire un circuit imprimé nous décidons de réaliser un circuit avec une plaque de bois de 3mm d'épaisseur. Nous avons fait un croquis des dimensions qu'il fallait puis nous avons réalisé la pièce à l'aide du logiciel CorelDraw. Nous lui rajoutons ensuite les trous des résistances puis nous découpons la pièce à l'aide de la découpeuse laser. Nous avons ensuite soudé le tout et nous avons obtenu ceci :



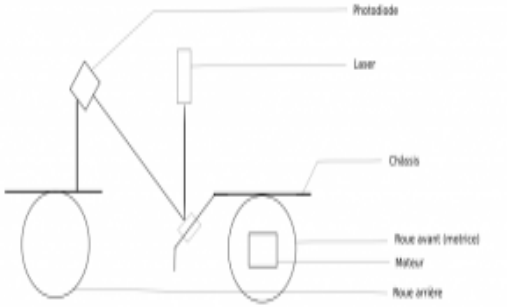


VII- Motorisation:

Il s'agit maintenant de déplacer l'intégralité du système comme s'il s'agissait d'une voiture télécommandée. Pour cela nous avons pensé à utiliser un châssis plat en dessous du quel nous aurions placé 4 roues. Nous aurions alors placé un moteur sur le châssis et aurions piloté les roues avant par Arduino en utilisant un système de tige filetée en guise de vis sans fin. Cependant pour obtenir une trajectoire bien rectiligne, il aurait probablement fallu utiliser un système de différentiel des roues. Nous avons alors pensé à installer trois roues et non pas quatre sur notre système en plaçant une seule roue à l'avant que nous motoriserons. Nous sommes alors aller discuter avec Laurent Teixeira pour avoir son avis sur cette solution, il nous a conseillé de ne pas utiliser de vis sans fin mais de raccorder le moteur directement à la roue avant motrice, en positionnant le moteur sous le châssis. En effet la vitesse de translation finale souhaitée étant très faible, cette solution est bien adaptée. De plus, nous avons pensé à un problème de répartition des masses. En effet, le moteur possède une masse conséquente par rapport à celle de notre système finale, ainsi en le plaçant à l'avant ou sur un des côtés du moteur, celui ci serait déséquilibré et risquerait de ne plus suivre une trajectoire bien rectiligne lors de sa mise en mouvement, ce qui nuirait ainsi considérablement à l'exploitation des données et donc à la précision de notre système. Il faudra donc placer une masse environ égale à celle du moteur à l'opposé de celui-ci, de manière à rendre notre système le plus homogène possible d'un point de vue massique.

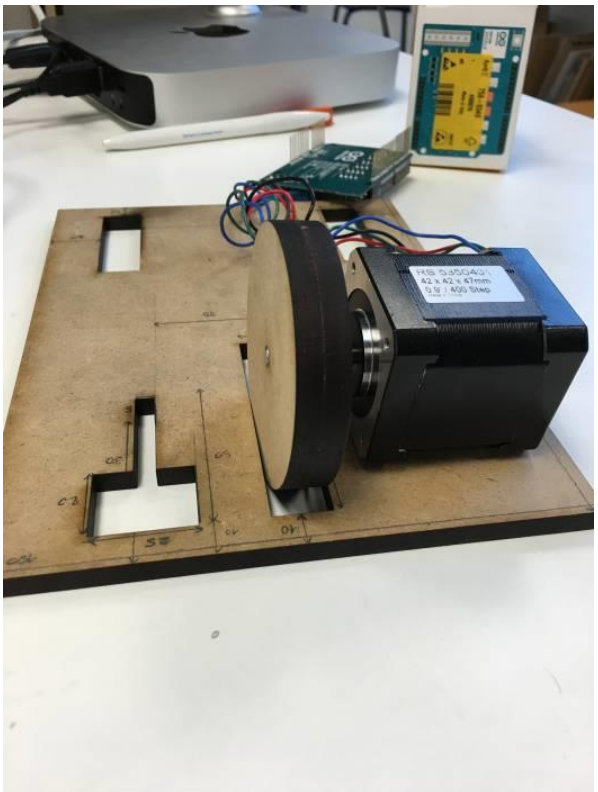


Vue du dessous



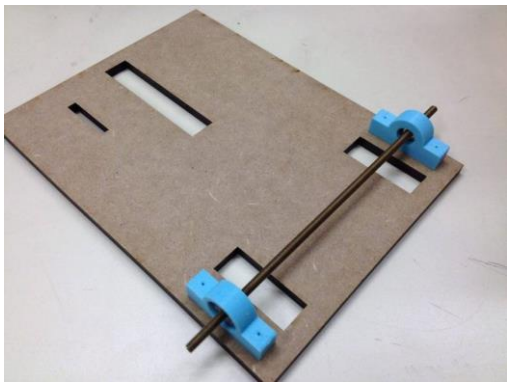
Vue de côté

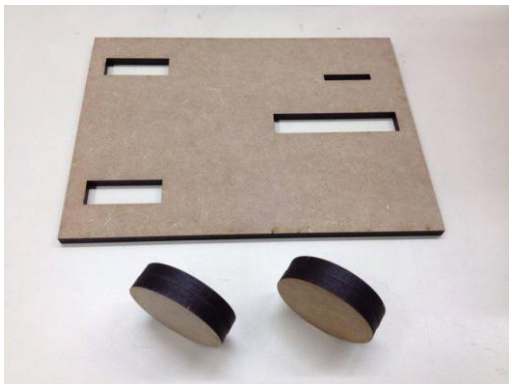
Cependant, d’après les calculs sur les dimensions du profilomètre, nous devons modifier ce qui était prévu, à savoir le positionnement du moteur. Ce dernier à la forme d'un cube de 42.3mm d'arrête tandis que pour une bonne exploitation des informations fournies par la tête de lecture, le châssis doit se trouver à environ 1cm du sol. Nous allons donc devoir placer le moteur au-dessus du châssis.
En ce qui concerne le couplage du moteur à la roue : la tige du moteur est cylindrique et lisse et il faut la relier de manière solide à la roue pour éviter tout patinage entre les deux. Deux solutions sont possibles : nous pouvons réaliser la roue en trouant son centre en forme de demi-cercle et limer le moteur de sorte à lui donner à peu près la même forme. Toutefois, compte tenu du faible couple que le moteur va exercer sur la roue et de la difficulté à limer correctement l’axe moteur nous avons choisi une autre solution : le diamètre de l'axe du moteur est de 5mm, nous allons donc réaliser la roue avec un trou au centre d'un diamètre de 4,8mm. Pour commencer, nous réalisons donc la roue avant qui est différente des 2 roues arrière. En effet, la position de l'axe du moteur est à prendre en compte et nous contraint de réaliser la roue avec un rayon plus grand que celle des petites roues. En réalisant les mesures, nous trouvons qu'il faut la réaliser avec un diamètre de 66mm. Nous découpons donc la roue à la découpeuse laser et nous limons légèrement l'axe du moteur pour qu'il agrippe mieux lorsqu'il sera dans la roue. A partir de là, nous rentrons en force l'axe du moteur dans le trou de la roue prévu à cet effet. C'est serré et il n'y a aucun glissement.



VIII- Conception du châssis et des roues arrière :

Après avoir effectué tous les calculs théoriques ; qui nous ont permis de savoir qu'il faut placer verticalement les photodiodes à environ 8cm du sol et horizontalement à environ 14cm du miroir (point de réflexion) ; nous pouvons passer à la réalisation du châssis. Nous nous occupons donc de la conception du châssis à la découpeuse laser, ainsi que des roues arrière. Pour faire tourner les roues nous avons choisi de les mettre sur roulement.



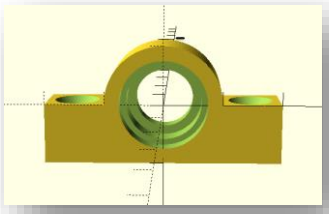


Nous avons collé les 4 cylindres deux par deux pour ne former que 2 roues plus épaisses et nous les avons limées pour éviter que le châssis vibre une fois les roues fixées dessus. En parallèle, nous avons redécoupé le châssis pour permettre aux roues d'entrer entièrement et nous avons fait un plus grand espace pour placer la tête de lecture. Nous avons ensuite percé les roues pour les faire passer dans la tige métallique et nous les avons fixées à celle-ci à l'aide d'écrous. Nous avons finalement placé les roulements aux extrémités de la tige et nous les avons fixés de la même manière. Après un test sur une surface plane, nous avons remarqué que les roues vibraient beaucoup trop et que cela allait avoir un impact néfaste sur la "lecture" de la surface. Nous avons donc utilisé des élastiques plats pour l'adhérence des roues.

IX- Réalisation des différentes pièces 3D :

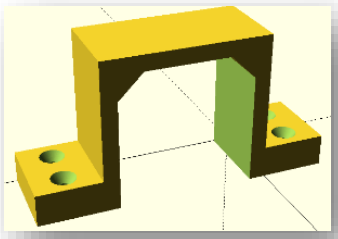
- Roulements :

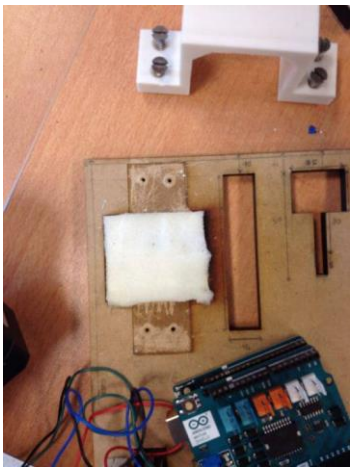
Les supports des roulements ont fendu après les avoir cloués, et une des attaches du miroir sur la tige a aussi légèrement fendu. Nous avons donc modifié le support des roulements afin de le visser et non de le clouer. La vis rentre donc parfaitement sans imposer trop de contraintes au plastique.



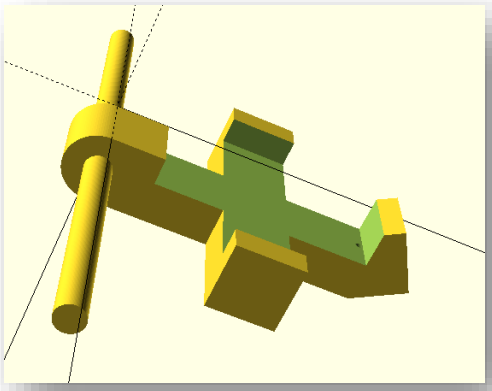
- Fixation pour le moteur :

Tout d'abord nous remarquons que le moteur crée d'importantes vibrations si nous le faisons tourner lentement, or il faut les limiter au maximum pour que notre aiguille ne les détecte pas et que ça ne fausse pas nos mesures. Pour cela nous décidons d'utiliser de la mousse très aérée et molle, que nous mettrons entre le châssis et le moteur. Nous avons aussi créé la pièce qui viendra plaquer le moteur contre le châssis afin qu'il ne bouge plus.





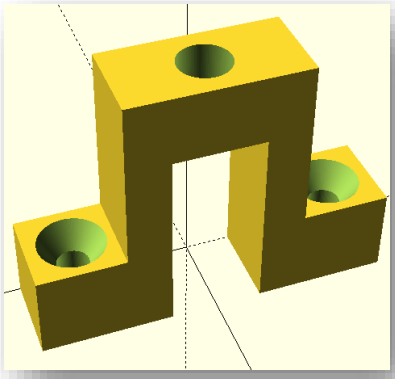
- Tête de lecture :

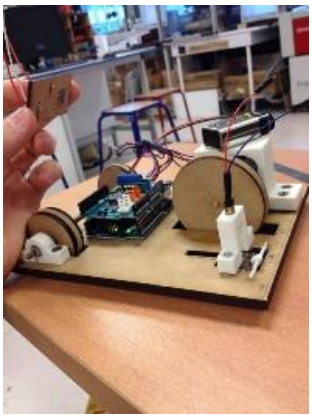


- Fixation du laser :

Nous avons créé le support laser sur Openscad. Une fois imprimée nous accrochons la pièce au châssis, insérons le laser dedans. Tout est parfait.

Il ne reste plus qu'à réaliser le support pour les photodiodes afin de les fixer au châssis, ainsi que de fixer le contrepoids et de faire les premiers tests.



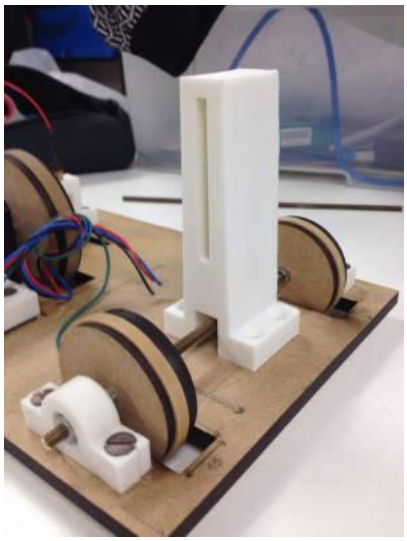


- Fixe photodiode :

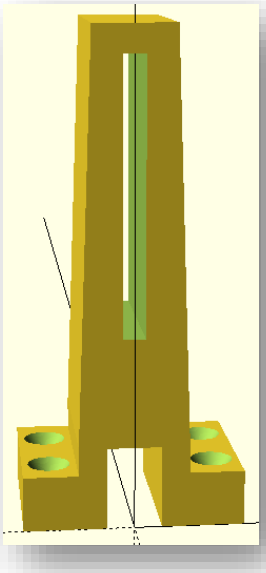
Tout d'abord il faut savoir que les calculs nous ont servi à avoir une idée de la taille de notre châssis et une vague idée de la distance miroir/photodiodes, mais qu'en réalité ça ne sera pas exactement juste puisque nous avons beaucoup d'incertitude (tige peut-être légèrement plus longue etc.). Il faut donc pouvoir faire un support amovible, c'est à dire qu'une fois monté au châssis, nous pourrons le configurer pour que le faisceau laser tape exactement au milieu des photodiodes avec un angle de 90°. Nous décidons que le

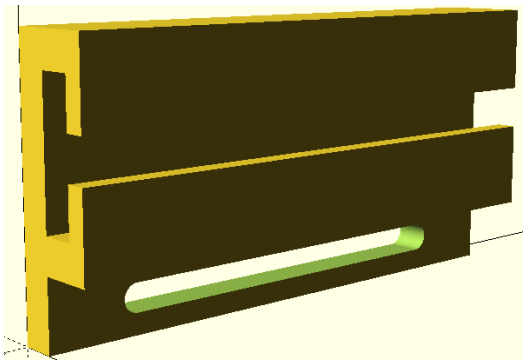
support soit amovible pour également pouvoir changer la résolution de notre profilomètre, en effet plus nous approcherons les photodiodes du miroir, plus nous pourrons détecter de grosses variations (de l'ordre du millimètre), et plus nous l'éloignerons plus nous pourrons détecter de petites variations. Nous avons donc réfléchi longuement pour savoir comment faire ce support.

Nous avons finalement décidé de le faire en 3 parties. La première est une sorte de mat, accroché au châssis, d'une taille d'environ 10cm (pour se laisser une marge d'erreur par rapport aux calculs), qui comporte une fente pour pouvoir régler la hauteur de la 2ème pièce ainsi que l'orientation. La 2ème pièce est en fait une sorte de raille où nous pourrons coulisser la 3ème pièce (celle qui tient les photodiodes), pour pouvoir avancer et reculer les photodiodes face au miroir. Nous créons donc les pièces sur Openscad, apportons quelques modifications et lançons les impressions.

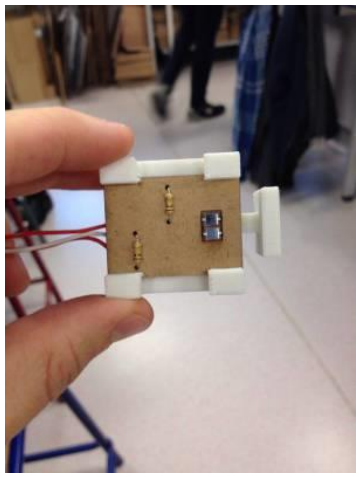


1

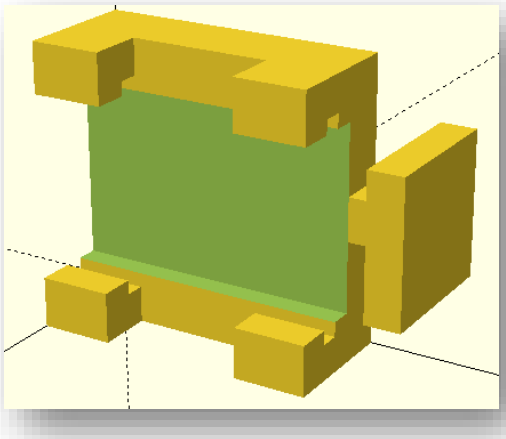


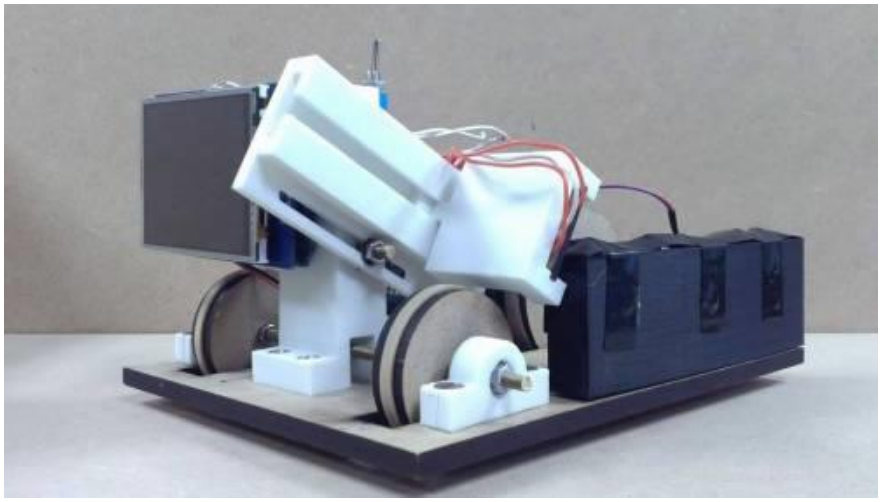


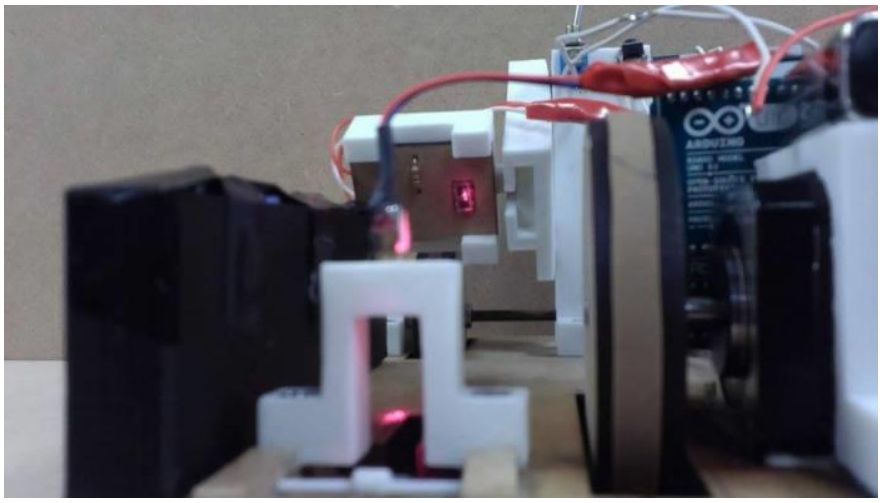
2



3

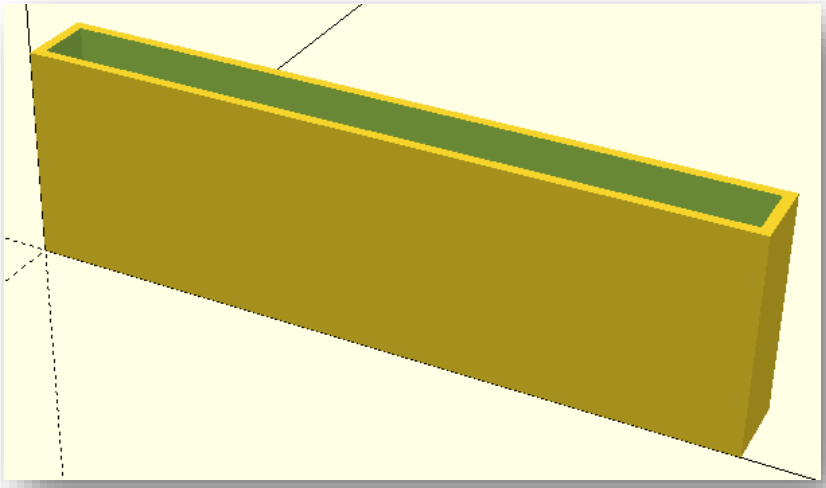






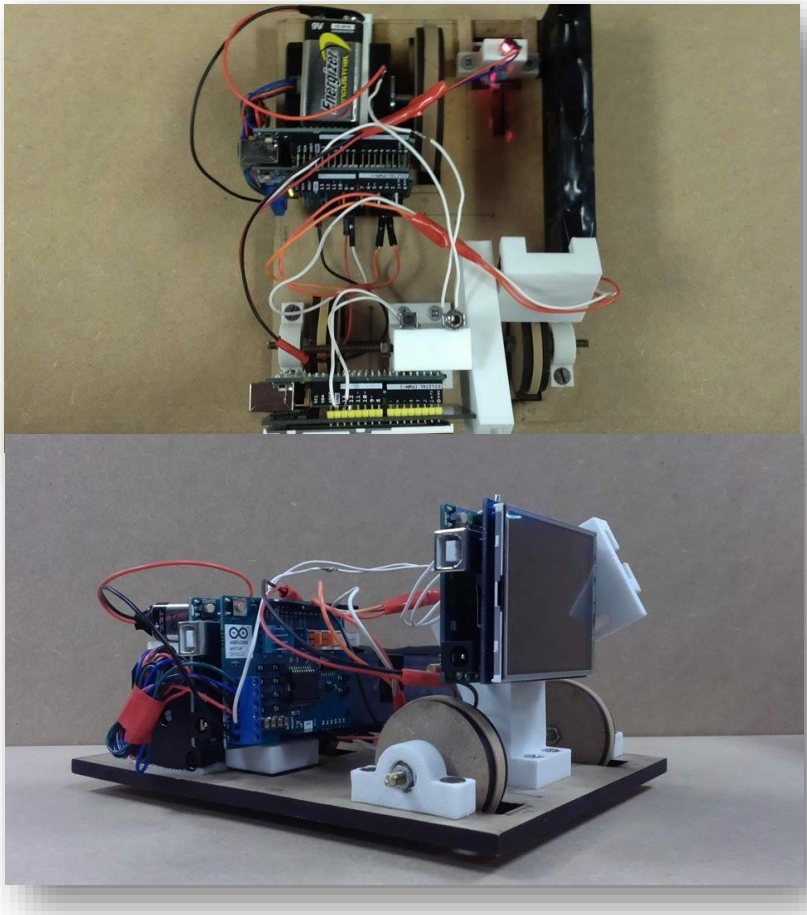
- <u>Réalisation du contrepoids</u> :

Pour équilibrer notre système, il nous faut une masse de 350g. Nous pensons donc à réaliser à l'imprimante 3D une pièce parallélépipédique qui se situera à droite du châssis (dans le sens de la marche) dans lequel nous pourrons y mettre diverses pièces permettant de réaliser le contrepoids. Nous décidons d'utiliser de petites vis (référence : M4*16 POZI). Nous dimensionons le boîtier : celui-ci fera 135x41x15 mm avec une épaisseur de 2mm.



X- Fixation de tous les composants :

Pour que notre rendu final soit propre, nous décidons de fixer tous les composants à la colle à chaud. Nous rajoutons également des boutons on/off. Le résultat obtenu est celui-ci :



XI- Programmation :

Nous avons fait de nombreux programmes sur Arduino, des programmes de tests, des programmes spécifiques (motorisation), et enfin des programmes finaux. Nous mettrons en annexes que les programmes que nous avions pour la présentation de notre projet. Je vous rappelle que nous avons commencé la programmation de l'écran LCD trop tard et nous nous sommes rendu donc compte trop tard qu'il nous fallait une carte Arduino MEGA, le shield + l'écran + les photodiodes, prennent trop de pins pour une carte Arduino UNO. Le premier programme en annexe sera donc celui présent dans la carte qui est relié au shield moteur (celle de devant). Il fait avancer le moteur, lit les valeurs renvoyées par les photodiodes, et les envoie à l'ordi via la communication série. Le 2ème programme est celui de la carte reliée à l'écran, il fait afficher une page de présentation, puis attend qu'on appuie sur un bouton pour lancer l'affichage d'une courbe (ici une fonction créneaux qu'on lui a demandé d'afficher). Enfin le dernier programme est celui qui relie les 2 premiers programmes entre eux (si on avait une carte Arduino MEGA).

XII- Cout du projet :

- Carte Arduino Uno + cordon : 19,50 €
- Diode laser : <5 €
- 2 photodiodes : <1 €
- Ecran TFT : 9 €
- Un moteur : 57 €
- Un shield moteur : 24 €

Total : 115,5 € (sans compter le coût des impressions 3D)

XIII- Conclusion :

Tous les composants de notre profilomètre fonctionnent séparément. Je vous invite à faire un tour sur le wiki à la fin de cette page : http://www.pmclab.fr/wiki/dokuwiki/doku.php?id=wiki:projets:telemetre:jdb. Vous y trouverez les tests de chacun des composants en vidéo.

En conclusion nous avons réalisé un profilomètre “ou presque”. C'est à dire que nous détectons des variations qui peuvent être très petites, de l'ordre d’une dizaine de micron (l'épaisseur d'un cheveux). Mais à cause des vibrations trop importantes du moteur, nous ne sommes pas assez précis. Nous n'avons également pas un affichage de la surface sur un écran LCD puisque nous avons programmé trop tard l'écran et nous nous sommes donc rendu compte trop tard qu'il nous fallait une carte Arduino MEGA. Nous sommes quand même fiers du résultat que nous avons obtenu. En effet, nous avons démarré de rien, avec aucune ressource sur internet, nous avons dû tout créer de nous-même et nous avons quand même réussi à obtenir un résultat.

En améliorations possibles ? Si nous repartions du début, nous ne ferions peut-être pas un profilomètre mobile, c'est à dire que le profilomètre serait fixe et que c'est la surface qui bougerait sur lui et non pas l'inverse, nous aurions comme ça plus besoin d’un moteur. Ou alors si nous gardions notre profilomètre mobile, nous changerions de moteur, pour qu'il n'y plus aucune vibration. Enfin comme nous l'avons déjà dit, nous prendrions une carte Arduino MEGA dès le début.

En conclusion de groupe, nous avons eu du mal à se mettre à notre projet dès début, mais nous avons ensuite bien avancé. Nous avons été un peu découragé à la fin d’avoir passé autant de temps sur le projet, et pour finalement ne pas être arrivés au résultat que nous souhaitions. Nous avons quand même découvert et appris de nombreuses choses, qui nous seront surement très utiles dans notre futur d’ingénieur.

Vous trouverez le tutoriel pour réaliser notre projet à cette page : http://www.pmclab.fr/wiki/dokuwiki/doku.php?id=wiki:projets:telemetre:tutoriel.

Temps passé sur le projet :

Par personne :

- Julien : 60h
- Théo : 85h
- Malissa : 20h
- Tristan : 21h
- Patrick : 13h
- Simon : 13h

En tout :

- 100h

Nous voudrions remercier :

- Le staff du fablab qui a toujours répondu à nos questions même si nous ne nous sommes pas servis de la totalité de leurs réponses.
- Laurent Teixeira pour ses conseils sur la motorisation.
- L'homme mystère de passage au fablab qui nous a conseillé sur le système de détection optique.
- « JoaoLopesF » pour la modification de la librairies d'adafruit afin qu'elle marche sur les écrans très bas prix. (http://www.instructables.com/id/How-to-use-24-inch-TFT-LCD-SPFD5408-with-Arduino-U/)
- Nos professeurs Christian Simon et Vincent Dupuis pour nous avoir suivis et conseillé tout au long de l'ARE.

Annexes :

Programme_final_sans_lcd §

```
void setup() {

  //pins de direction
  pinMode(12, OUTPUT); //CH A
  pinMode(13, OUTPUT); //CH B
  //pins de frein
  pinMode(9, OUTPUT); // CH A
  pinMode(8, OUTPUT); // CH B

  pinMode(A2, INPUT); //photodiode 1 en A0
  pinMode(A3, INPUT); //photodiode 2 en A1
  Serial.begin(9600); //initialisation de la comunication avec le PC
  analogReference(INTERNAL); // valeur de référence analogique de 0 à 1,1V

}

void loop() {

  digitalWrite(9, HIGH);  //On enlève le frein sur CH A
  digitalWrite(8, LOW); //On le met sur CH B
  digitalWrite(12, LOW); // on choisis la direction
  analogWrite(3, 200);   //on fait tourner CH A
  delay(temp_m);
  detection();

  digitalWrite(9, LOW);  //On met le frein sur CH A
  digitalWrite(8, HIGH); // On l'enlève sur CH B
  digitalWrite(13, HIGH);  //On choisis la direction
  analogWrite(11, 200);   //On fait tourner CH B
  delay(temp_m);
  detection();

  digitalWrite(9, HIGH);  //On enlève le frein sur CH A
  digitalWrite(8, LOW); //On le met sur CH B
  digitalWrite(12, HIGH); // on choisis la direction
  analogWrite(3, 200);   //on fait tourner CH A
  delay(temp_m);
  detection();

  digitalWrite(9, LOW);  //On met le frein sur CH A
  digitalWrite(8, HIGH); // On l'enlève sur CH B
  digitalWrite(13, LOW);  //On choisis la direction
  analogWrite(11, 200);   //On fait tourner CH B
  delay(temp_m);
  detection();

} void detection() {

  valeur1 = analogRead(A2);
  valeur2 = analogRead(A3); //lecture signaux
  delay(attente);
  valeur_fin = valeur1 - valeur2; //luminosité
  Serial.println(valeur_fin); //renvoie de la valeur sur le moniteur série
  delay(attente);
}
```

sketch_apr27a §

```ino
#include <SPFD5408_Adafruit_GFX.h>
#include <SPFD5408_Adafruit_TFTLCD.h>
#include <SPFD5408_TouchScreen.h>

#define BLACK 0x0000
#define BLUE 0x001F
#define RED 0xF800
#define GREEN 0x07E0
#define CYAN 0x07FF
#define MAGENTA 0xF81F
#define YELLOW 0xFFE0
#define WHITE 0xFFFF

Adafruit_TFTLCD tft(A3, A2, A1, A0, A4);

float y1, y0;
int hauteur = tft.width();
int largeur = tft.height();

void setup() {

  pinMode(13, INPUT);
  tft.reset();
  tft.begin(0x9341);
  tft.setRotation(3);

//affichage nom projet
  tft.fillScreen(BLACK);
  tft.setTextColor(BLUE);
  tft.setTextSize(4);
  tft.setCursor(20, 90);
  tft.print("Pr");
  tft.setTextColor(CYAN);
  tft.print("of");
  tft.setTextColor(GREEN);
  tft.print("il");
  tft.setTextColor(YELLOW);
  tft.print("om");
  tft.setTextColor(MAGENTA);
  tft.print("et");
  tft.setTextColor(RED);
  tft.println("re");
  tft.setTextSize(2);
  tft.setTextColor(WHITE);
  tft.setCursor(60, 135);
  tft.print("Ou presque ...");
  delay(5000);
  tft.fillScreen(BLACK);
  while (digitalRead(13) == 1){
    tft.fillScreen(BLACK);
    tft.setTextColor(WHITE);
    tft.setCursor(65, 100);
    tft.print("Restez appuye sur");
    tft.setCursor(95, 130);
    tft.print("le bouton");
    delay(1000);
}}
void loop() {

tft.fillScreen(BLACK);

//affichage des axes et des graduations
tft.drawLine(20, 10, 20, 230, WHITE);
tft.drawLine(20, 120, 310, 120, WHITE);
for (int x=20; x < largeur; x+=20){
  tft.drawLine(x, 115, x, 125, WHITE);
}
for (int y = 20; y < hauteur; y+=20){
  tft.drawLine(15, y, 25, y, WHITE);
}

//traçage de la courbe et affichage des valeurs
y1 = 0;
for (float i=0; i < largeur; i+=4){
  y0 = y1;
  //y1 = 2*sin(i/10) * cos(i/7) * 80 ;
  y1 = (sqrt(sin(i/10)*sin(i/10)) + sin(i/10))*80/ 2*sin(i/10);
  tft.drawLine(i, y0 + 80, i+4, y1 + 80, CYAN);
  tft.setCursor(150, 0);
  tft.setTextColor(CYAN);  tft.setTextSize(2);
  tft.print("Valeur: ");
  tft.print(float(120 - (y1 + 80)));
  delay(100);
  tft.fillRect(230, 0, 340, 20, BLACK);
}}
```

programme_final_MEGA.ino

```
//inclusion des librairies
#include <SPFD5408_Adafruit_GFX.h>
#include <SPFD5408_Adafruit_TFTLCD.h>
#include <SPFD5408_TouchScreen.h>

//définition des couleurs
#define BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF

//définition des pins ou sont brancher les broche de l'ecran
Adafruit_TFTLCD tft(A3, A2, A1, A0, A4);

//définition des variables
float y1, y0;
int hauteur = tft.width(), largeur = tft.height();
int valeur1, valeur2, valeur_fin;
int attente = 50;
int temp_m = 500 - attente;

void setup() {

  //pins de direction
  pinMode(12, OUTPUT); //CH A
  pinMode(13, OUTPUT); //CH B
  //pins de frein
  pinMode(9, OUTPUT); // CH A
  pinMode(8, OUTPUT); // CH B

  pinMode(A7, INPUT); //photodiode 1 en A7
  pinMode(A8, INPUT); //photodiode 2 en A8
  analogReference(DEFAULT); // valeur de référence

  pinMode(31, INPUT); //pin du bouton de l'ecran (A choisir)
  tft.reset();
  tft.begin(0x9341);
  tft.setRotation(3); //on met l'ecran dans le bon sens

  //affichage du nom de notre projet
  tft.fillScreen(BLACK);
  tft.setTextColor(BLUE);
  tft.setTextSize(4);
  tft.setCursor(20, 90);
  tft.print("Pr");
  tft.setTextColor(CYAN);
  tft.print("of");
  tft.setTextColor(GREEN);
  tft.print("il");
  tft.setTextColor(YELLOW);
  tft.print("om");
  tft.setTextColor(MAGENTA);
```

Nouveau

programme_final_MEGA.ino

```
  tft.setTextColor(MAGENTA);
  tft.print("et");
  tft.setTextColor(RED);
  tft.println("re");
  tft.setTextSize(2);
  tft.setTextColor(WHITE);
  tft.setCursor(60, 135);
  tft.print("Ou presque ...");
  delay(5000);
  tft.fillScreen(BLACK);

  //tant que le bouton n'est pas enclenché
  while (digitalRead(31) == 1){
    tft.fillScreen(BLACK);
    tft.setTextColor(WHITE);
    tft.setCursor(65, 100);
    tft.print("Restez appuye sur");
    tft.setCursor(95, 130);
    tft.print("le bouton");
    delay(1000);
  }
}

void loop() {
  valeur1 = analogRead(A7);
  valeur2 = analogRead(A8); //lecture signaux
  delay(attente);
  y1 = valeur1 - valeur2; //luminosité
  tft.fillScreen(BLACK);

  //affichage des axes et des graduations
  tft.drawLine(20, 10, 20, 230, WHITE);
  tft.drawLine(20, 120, 310, 120, WHITE);
  for (int x=20; x < largeur; x+=20){
    tft.drawLine(x, 115, x, 125, WHITE);
  }
  for (int y = 20; y < hauteur; y+=20){
    tft.drawLine(15, y, 25, y, WHITE);
  }
  //traçage de la courbe et affichages des valeurs
  for (int i=0; i < largeur; i++)//boucle qui va décaler d'un pixel
  {
    y0 = y1; //y0 prend l'ancienne valeur
    valeur1 = analogRead(A7);
    valeur2 = analogRead(A8); //nouvelle valeur
    delay(attente);
    y1 = valeur1 - valeur2; //luminosité
    tft.drawLine(i, y0 - 100, i+1, y1 - 100 , CYAN); //on trace
    tft.setCursor(160, 0);
    tft.setTextColor(CYAN);  tft.setTextSize(2);
    tft.print("Tension: "); //On affiche la tension associé
    tft.print(y1*5/1023);
    delay(100);
    tft.fillRect(260, 0, 340, 20, BLACK);

    //On fait avancer d'un pas
```

```
    delay(100);
    tft.fillRect(260, 0, 340, 20, BLACK);

    //On fait avancer d'un pas
    /* /!\/!\ Numéro des pins à modifier /!\/!\ */
    if (i%4 == 0) {
      digitalWrite(9, HIGH);  //On enlève le frein sur CH A
      digitalWrite(8, LOW); //On le met sur CH B
      digitalWrite(12, LOW); // on choisis la direction
      analogWrite(3, 200);   //on fait tourner CH A
      delay(temp_m);
    }
    else if(i%4 == 1) {
      digitalWrite(9, LOW);  //On met le frein sur CH A
      digitalWrite(8, HIGH); // On l'enlève sur CH B
      digitalWrite(13, HIGH);  //On choisis la direction
      analogWrite(11, 200);   //On fait tourner CH B
      delay(temp_m);
    }
    else if(i%4== 2) {
      digitalWrite(9, HIGH);  //On enlève le frein sur CH A
      digitalWrite(8, LOW); //On le met sur CH B
      digitalWrite(12, HIGH); // on choisis la direction
      analogWrite(3, 200);   //on fait tourner CH A
      delay(temp_m);
    }
    else {
      digitalWrite(9, LOW);  //On met le frein sur CH A
      digitalWrite(8, HIGH); // On l'enlève sur CH B
      digitalWrite(13, LOW);  //On choisis la direction
      analogWrite(11, 200);   //On fait tourner CH B
      delay(temp_m);
    }
  }
}

Compilation terminée.

Le croquis utilise 26 350 octets (10%) de l'espace de stockage de proc
Les variables globales utilisent 227 octets (2%) de mémoire dynamique,
```