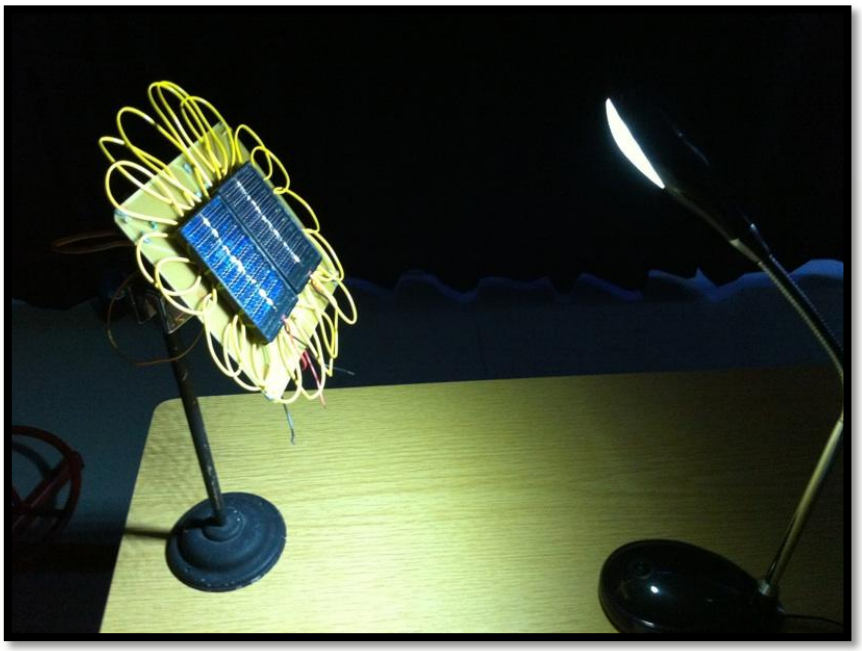
P6 – Projet de physique
Sujet 12 : Le tournesol photovoltaïque
STPI 2

Rapport de projet :

TOURNESOL PHOTOVOLTAÏQUE



Etudiants :

François DECQ **Baptistin ROUSSEL**
Marie LEMAGNANT **Nelcis ZORA**

Responsable du projet :

Corentin JOUEN


INSA
ROUEN

Date de remise du rapport : ***16/06/2014***

Référence du projet : ***STPI/P6/2014 – N°12***

Intitulé du projet : ***Tournesol Photovoltaïque (Suivi solaire par panneaux photovoltaïques)***

Type de projet : ***Biblio - Expérimentation***

Objectifs du projet :

Ce projet avait pour but de réaliser un tournesol électronique (ou tracker solaire). Nous avons découvert que cela était réalisable à l'aide d'un microcontrôleur Arduino MEGA 2560 et de deux servomoteurs. En quelques mots, notre montage devait être capable de se déplacer en suivant la lumière, quel qu'elle soit. A notre échelle, nos dispositifs d'éclairement se résumaient à des lampes torche voire lumières de portable, plutôt que la lumière solaire.

En plus du côté expérimental, ce projet devait nous permettre de nous initier à la conduite en groupe de projets, de passer du théorique à l'expérimental et du côté humain, d'acquérir de l'autonomie.

Mots-clefs du projet *(4 maxi)* : ***innovation- énergie– capteur – électronique***

Table des matières

Introduction

Notre formation d'ingénieur nous prépare à occuper des fonctions scientifiques ou techniques en vue de prévoir, créer, organiser, diriger et contrôler les travaux qui en découlent. C'est dans cette optique de conception et de direction de projets que nous avons abordé le projet de P6.

Par groupe de 4 étudiants et pendant 14 semaines, nous nous sommes retrouvés les mercredis en début d'après-midi pendant les 1h30 de séances ainsi que les lundis ou mardis midi hors séance, afin d'avancer plus vite et définir une répartition des tâches pour la séance du mercredi. Nous étions encadrés par un professeur dont la fonction était de suivre l'avancement de notre projet (lors des séances et par les comptes-rendus hebdomadaires) et de nous aider en cas de besoin.

A l'aide d'un microcontrôleur Arduino MEGA 2560, de capteurs solaires, de panneaux photovoltaïques, de deux servomoteurs, nous sommes parvenus à réaliser un tournesol électronique (ou tracker solaire). Pour la plupart d'entre nous, ce fut dès le début une grande découverte, notamment pour les composés automatiques et le langage de programme Arduino.

Tout d'abord, dans la première partie de ce rapport, nous présentons ce qu'est un tournesol photovoltaïque. Puis, dans un second temps, nous expliquerons selon quelle méthodologie nous avons procédé et comment avons-nous organisé notre travail. Enfin, nous terminerons ce rapport par l'analyse des travaux réalisés, des montages aux tests, et des résultats obtenus.

I. Tournesol électronique

1) L'énergie solaire photovoltaïque

L'énergie solaire photovoltaïque est une énergie électrique renouvelable, produite par les rayonnements du Soleil. La production de cette énergie peut se faire grâce aux cellules photovoltaïques, composant électrique qui, exposé à la lumière, produit de l'électricité en quantité proportionnelle à la puissance lumineuse incidente.

Un ensemble de cellules photovoltaïques forme un panneau solaire photovoltaïque qui délivre une puissance qui est de l'ordre de 100 à 200 W/m². Mais l'énergie réellement capté par ce panneau solaire dépend de plusieurs facteurs : la latitude, la saison, l'heure de la journée, la météo, etc. L'électricité produite est disponible sous forme d'électricité directe ou stockée en batteries (énergie électrique décentralisée) ou en électricité injectée dans le réseau.

En France, la puissance totale raccordée est de 4,673 GW fin 2013 (France métropolitaine et DOM). La part de l'électricité photovoltaïque dans la production totale d'électricité est de 0,66% sur l'année 2013.

2) Le tracker solaire

Dans notre cas le tracker solaire résout le problème de perte angulaire.

Qu'est-ce qu'un tracker solaire ? Le tracker solaire est une structure portante qui permet d'orienter les panneaux solaires photovoltaïques en fonction de la position du Soleil, afin d'en augmenter la productivité.

En effet, le Soleil change de position tout au long de la journée, l'angle d'incidence des rayons avec les panneaux augmente et l'éclairage n'est donc pas optimal. C'est pourquoi nous nous basons sur le modèle du tournesol, qui suit le trajet du soleil afin d'accéder au meilleur ensoleillement possible pour assurer la photosynthèse.

L'idée est que le surplus d'énergie produite grâce à l'orientation des panneaux sera plus important que l'énergie dépensée à orienter les panneaux de façon optimale. En effet, un tracker solaire peut augmenter le rendement d'un panneau photovoltaïque jusqu'à 40 % de plus qu'un panneau fixe.

La meilleure orientation est celle qui fait que les rayons soient perpendiculaires aux panneaux.

Suivre le Soleil nécessite deux axes : en azimut (d'Est en Ouest à mesure de l'avancée de la journée) et en hauteur (selon la saison et l'avancée de la journée). La meilleure façon de suivre le Soleil est donc d'utiliser un tracker à deux axes, mais il en existe avec un seul axe qui dépend de l'azimut, l'angle des panneaux par rapport au sol étant calculé de façon optimum selon la latitude.

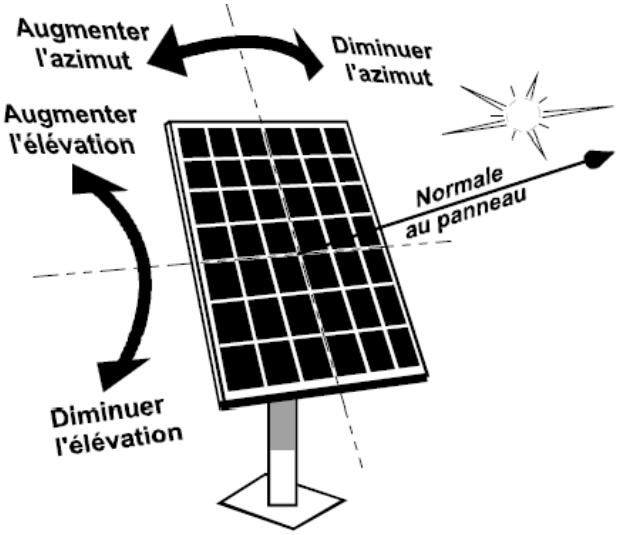


Figure 1. Tracker solaire à système bi-axial

Il existe deux principaux mécanismes de suivi :

- Le mécanisme d'horlogerie, qui consiste à programmer la position des panneaux grâce à des calculs permettant de déterminer l'orientation optimale en fonction de la latitude, la saison et l'heure en se basant sur des observations des années précédentes.
- Le système d'asservissement, qui consiste à toujours réduire l'écart entre l'ensoleillement maximum et l'ensoleillement perçu par les panneaux.

3) Le principe du tracker solaire avec Arduino

Notre réalisation d'un tracker solaire avec Arduino se base principalement sur un automate capable de suivre une source lumineuse en s'orientant de façon optimale en direction des rayons lumineux.

Les mouvements de rotations se produisant selon deux axes (vertical et horizontal) nous utilisons donc deux moteurs commandés par notre carte Arduino (le microcontrôleur).

La méthode utilisée est comparable à celle de l'asservissement. Notre panneau solaire est équipé de quatre capteurs photosensibles et il s'oriente en fonction de la luminosité reçue par chaque capteur. Notre programme compare la luminosité reçue par chaque capteur et ajuste la position si une différence est détectée. Par exemple, si les capteurs du haut reçoivent plus de lumière que ceux du bas, le programme va détecter cette différence et va orienter le panneau vers le haut pour compenser. Ce sera le même principe pour la gauche et la droite.

La programmation de la carte Arduino a été nécessaire pour que le montage puisse fonctionner sans connexion avec l'ordinateur, le code étant enregistré sur la carte Arduino.

Enfin nous avons fabriqué aussi une carte électronique, que nous utilisons comme support, ainsi nous n'avons pas besoin de câblage, les panneaux photovoltaïques et les capteurs sont directement montés sur cette carte électronique et le microcontrôleur Arduino se trouve en dessous de cette dernière

II. Méthodologie et organisation du travail

1) Méthodologie

A. Planning prévisionnel

Notre responsable de projet nous a demandé de réaliser un planning prévisionnel (voir page suivante) lors de la deuxième séance que nous avons essayé de suivre tout au long du semestre. Globalement, nous pouvons dire que nous avons mal estimé le temps que prendrait le projet en lui-même, hors extension.

En effet, le montage de la plaque s'est avéré être plus long que prévu. Ceci est certainement dû à notre manque de connaissance sur le logiciel KiCad, pour lequel notre responsable de projet a dû nous expliquer le fonctionnement, mais aussi aux quelques séances pour lesquelles notre responsable de projet a dû s'absenter, ne pouvant donc pas nous expliquer certains fonctionnements techniques dont nous avions besoin.

Finalement, nous n'avons donc pas pu travailler sur le rendement, le stockage de l'énergie, ni l'anémométrie.

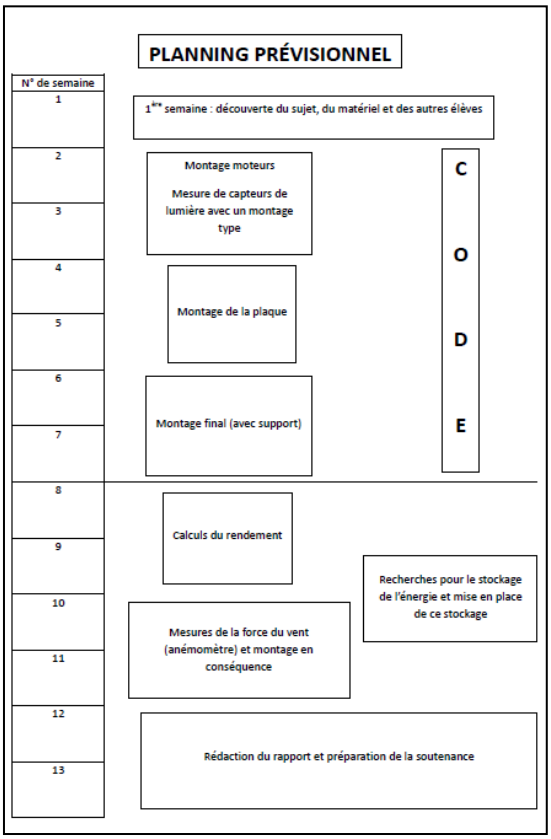


Figure 2. Planning prévisionnel

Les différentes étapes de notre projet ont été travaillées comme suit :

- **Montage servomoteurs** : séance 2, puis démontage/remontage dû au dysfonctionnement d'un des deux servomoteurs séances 7-8.
- **Mesures capteurs de lumière** : séances 2 à 6.
- **Tests moteurs** : séances 3-4, 7 et 9-10.
- **Montage de la plaque (KiCad)** : séances 3 à 7.
- **Codage** : séances 2 à 7 et 9.
- **Montage final** : séances 8 à 10.

B. Déroulement type d'une séance

Nous n'avions pas réellement de schéma type par séance, mais nous nous organisions instinctivement comme suit :

- Début de séance, résumé de l'avancement du projet.
- Résumé du travail à faire lors de la séance.
- Discussion du travail à faire avec notre responsable de projet.
- Séparation des tâches.
- Travaux séparés, si besoin avec l'aide de notre responsable de projet.
- Dix minutes avant la fin, retour sur le travail de chacun.
- Discussion du travail à faire lors de la prochaine séance.

2) Organisation du travail

Pour mener à bien notre projet de monter un héliostat solaire nous avons du nous organiser et répartir des rôles pour chaque personne du groupe. Au vu du travail, Il est vrai que le partage des taches était le meilleur moyen de parvenir à terminer ce projet conséquent.

Tout d'abord, nous pouvons parler des comptes rendus que nous devions rédiger après chaque séance de projet pour nous permettre de garder une trace de nos avancés mais aussi des difficultés rencontrées pendant ces séances. Nous avons donc décidé que chacun de nous écrirait à tour de rôle un compte rendu afin d'avoir une bonne répartition du travail. En plus du temps alloué à ce projet dans notre emploi du temps, nous avons décidés de nous réunir chaque semaine sur notre temps du midi. Ainsi nous pouvions avancer le projet tous ensemble et faire toutes les modifications nécessaire à notre rythme. Ce temps de travail nous a permis d'avancer beaucoup plus rapidement que si nous n'avions travaillé que sur les heures réservées.

En ce qui concerne notre travail sur les séances à proprement parler, nous nous répartissions les tâches en début de séance en nous fixant un objectif à atteindre avant la fin de celle-ci. Mais quand le travail que nous devions effectuer prenait plusieurs séances alors ceux qui s'en était chargé en premier lieu le poursuivait. Effectivement, c'est plus facile de suivre un travail du début à la fin plutôt que d'essayer de prendre le travail déjà débuté. Mais globalement nous nous sommes répartis les tâches de la façon suivante :

Le **montage des deux servomoteurs** a été assuré par François et Baptistin qui se sont aidés des manuels disponibles sur le site d'achat de matériel électronique « L'Impulsion ».

Le **montage du circuit avec les résistances, les capteurs solaires, la carte Aduino ainsi que les servomoteurs** à été assuré par Nelcis et Marie. Il nous a fallu un peu de temps avant que le montage soit parfaitement opérationnel.

Une fois ce montage réalisé, nous avons pu commencer à faire des tests pour voir si les acquisitions fonctionnaient correctement ou pour voir si les servomoteurs marchaient

convenablement. C'est l'une des seule parties que nous avons effectuée tous ensemble car nous étions heureux de constater que notre projet prenait forme.

Pour que notre projet arrive à son terme, il nous fallait un support qui nous permettrait d'accrocher le ou les panneaux solaires ainsi que les différents composants nécessaires au fonctionnement de l'héliostat. Baptistin puis Marie se sont donc chargés de dessiner **une carte électronique à l'aide du logiciel KiCad**.

Enfin, pour que le tournesol électronique fonctionne normalement, il fallait qu'il puisse suivre les sources lumineuses. Il fallait donc créer un **code informatique** qui fasse en sorte que le panneau suive la source lumineuse au plus près. Ce sont François et Nelcis qui se sont chargé du codage de ces fonctions.

III. Travail réalisé et résultats

1) Montages

Avant tout début de test et de codage, il nous a fallu monter chaque élément du système, c'est-à-dire : les servomoteurs, et le circuit électrique. Ces montages ont été réalisés sur les premières séances.

A. Montage des servomoteurs

François et Baptistin se sont occupés du montage des servomoteurs lors de la deuxième séance.

Pour cela ils ont dû télécharger la notice disponible sur le site limpulsion.fr.

Il est à noter que les servomoteurs ont été fournis par la société L'Impulsion. Le kit servomoteur était composé des deux servos standards, d'un ensemble d'écrous de vis, d'une tasse de roulement, et de deux parenthèses en aluminium, l'une en forme de U et l'autre multifonctionnelle.

Le montage s'est déroulé en 4 étapes :

- Dans un premier temps, la tasse de roulement est vissée sur la parenthèse multifonctionnelle.
- Puis, cette parenthèse est montée sur l'un des servomoteurs standard.
- Le deuxième servomoteur est fixé au premier assemblage.
- Enfin, la parenthèse U-forme est attachée au deuxième servomoteur.

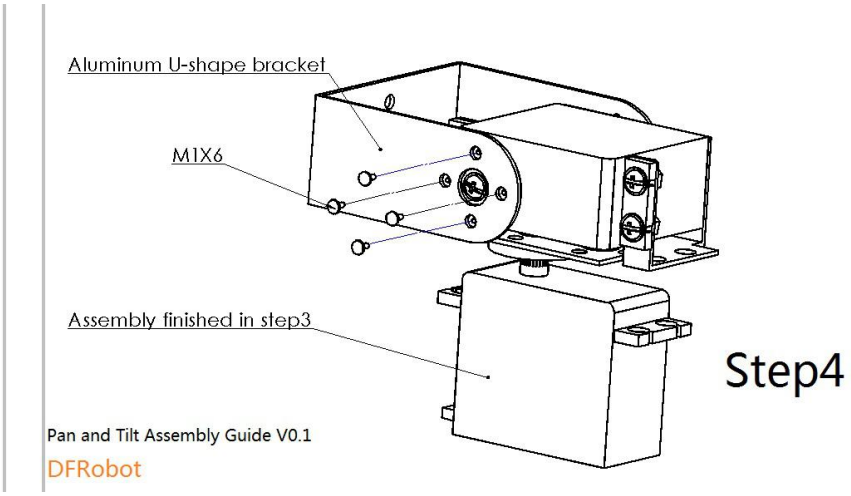


Figure 3. Etape 4 du montage, la parenthèse U-forme est vissée sur le deuxième servomoteur

B. Circuit électrique

Le montage du circuit électrique s'est fait sur la deuxième séance, pendant que François et Baptistin montaient les servomoteurs, Marie et Nelcis se sont occupées du circuit électrique.

Ce circuit allait servir à trouver la résistance idéale pour le montage dans un premier temps puis de tester les capteurs de lumière.

Elles ont d'abord effectué des recherches sur internet concernant la luminosité reçue en plein jour, puis grâce au conseil du maître de projet elles ont recherché la résistance nécessaire pour avoir une tension qui tournait autour de celle reçue en plein jour.

Avec le schéma électrique suivant, simplifié et représentant la tension reçue par le capteur de lumière R2 et la résistance R1, et grâce à la relation du pont diviseur de tension elles sont parvenue à trouver la résistance idéale pour le montage :

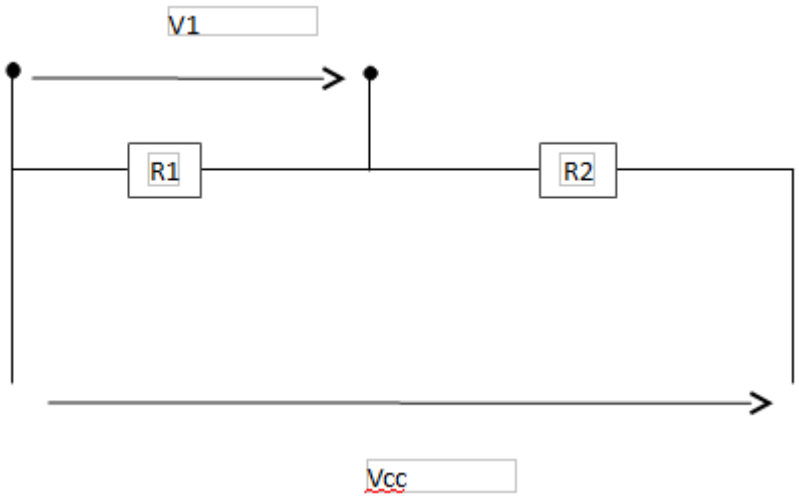


Figure 4. Schéma électrique simplifié du montage

On a: $\frac{Vcc}{R1+R2} = \frac{V2}{R2} \rightarrow V2 = \frac{R2}{R1+R2} \times Vcc$

Et on sait que V2 varie entre 2,5V (nuit) et 5V (jour) :

- Quand il fait jour : V2=5V=¿R2R1+R21
- Quand il fait nuit : V2=2.5V=¿R2R1+R21/2

Au final, on obtient R2=R11kΩ.

Après la partie calculatoire, il a fallu faire le branchage de tous les composants sur une platine d'expérimentation, on a aussi branché les deux servomoteurs, et la carte arduino afin de les tester :

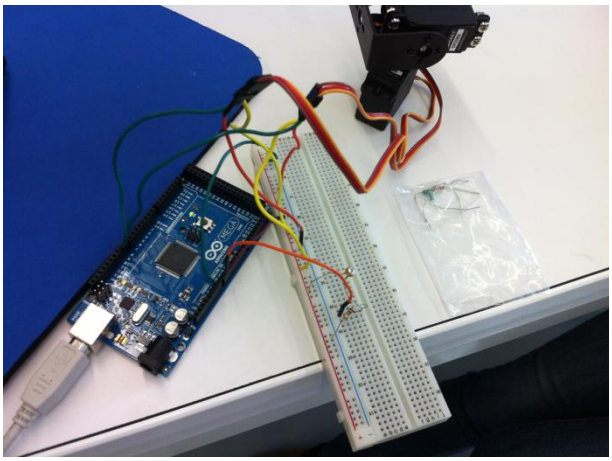


Figure 5. Montage électronique pour le test des composants

C. Support

La recherche du support s'est faite durant les dernières séances. Avec l'aide de la technicienne nous avons pu trouver un support adapté à notre projet.

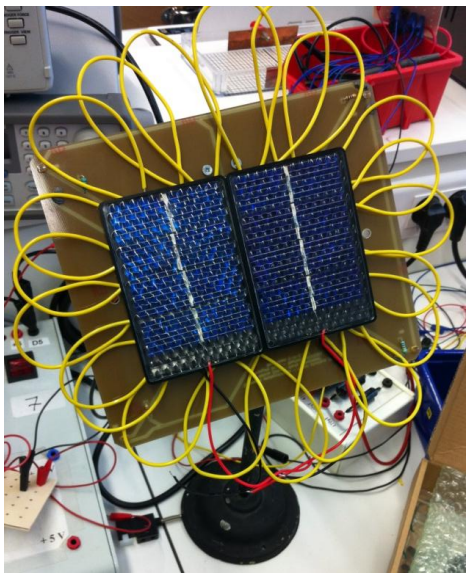


Figure 6. Montage final

Nous avons donc monté notre assemblage sur le support lors de la 10ème séance. Cependant, il se trouvait que le poids des panneaux photovoltaïque n'était pas adapté à la puissance des servomoteurs, celui-ci était trop important. Nous avons dû enlevé le panneau principal, en y gardant les deux petits.

Le montage final se composait alors du support, des servomoteurs, de la carte électronique, des capteurs de lumières, et des panneaux photovoltaïques.

2) Tests

A. Langage Arduino

Notre projet a été entièrement réalisé autour du langage Arduino. En effet, notre microcontrôleur est un Arduino MEGA 2560.

Avant ce semestre et ce projet, nous n'avions aucune connaissance de ce langage de programmation dérivé du C et du C++, langages pour lesquels nous n'avions eu aucune formation à l'INSA auparavant non plus. Il nous a donc fallu tout d'abord apprendre le fonctionnement de ce langage et sa syntaxe.

"Si on le considère généralement, Arduino est un circuit imprimé en matériel libre (les composants ne sont pas en logiciel libre) sur lequel se trouve un microcontrôleur qui peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des tâches très diverses comme la domotique, le pilotage d'un robot, etc... Arduino était à l'origine principalement destiné à la programmation multimédia interactive en vue de spectacle ou d'animations artistiques. Arduino peut être utilisé pour construire des objets interactifs indépendants, ou bien peut être connecté à un ordinateur pour communiquer avec ses logiciels. Des informations sont fournies pour ceux qui souhaitent assembler l'Arduino eux-mêmes." (Wikipédia : Arduino).

Nous avons donc été récupérer ses informations sur Internet, et plus spécifiquement celles de notre modèle d'Arduino, l'Arduino MEGA 2560.

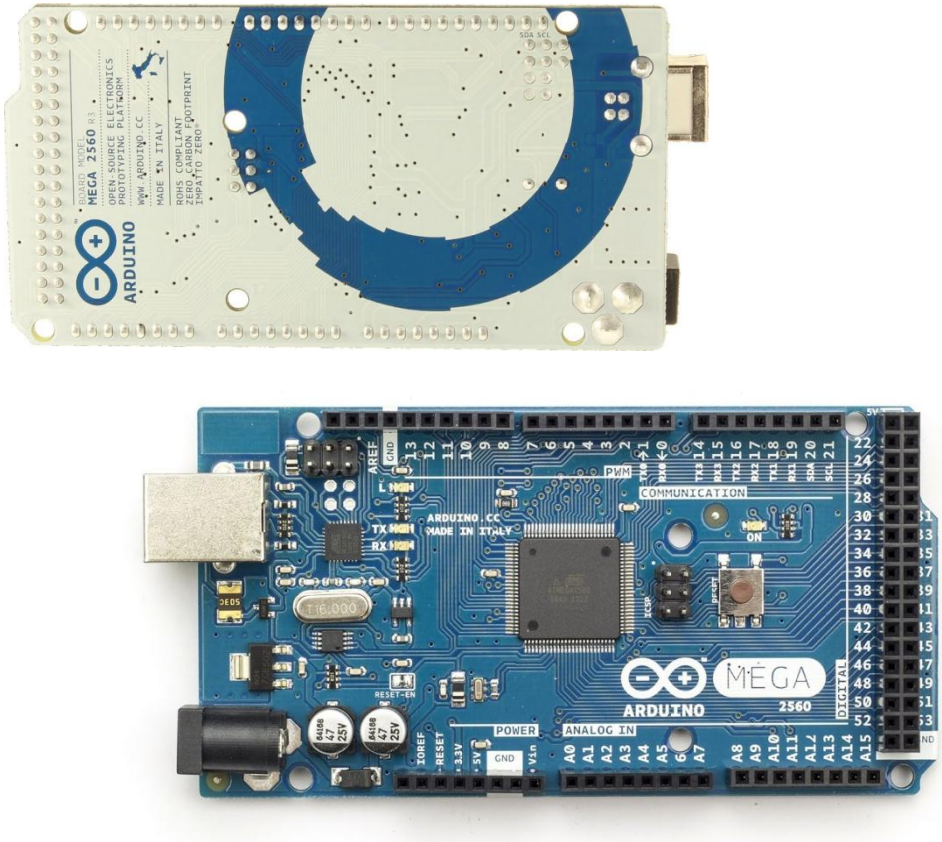


Figure 7. Microcontrôleur Arduino MEGA 2560

Ce modèle est un des plus puissants en terme de mémoire, puisque sa capacité totale est de 256 Ko. Il fonctionne sous une tension opérante de 5V, et il suffit de le connecter à un port USB pour qu'il se mette en marche.

B. Codage

Nous allons ici expliquer les différentes parties du code final, comment l'information de luminosité est récupérée, analysée puis comment il commande le mouvement des moteurs. Nous allons tout d'abord expliquer le fonctionnement global du code, et donc du montage, puis détailler ce code.

Nous disposons de quatre capteurs, que l'on a placés aux quatre coins de la carte électronique. L'information numérique récupérée par ces capteurs est acquise toutes les 10ms. Le code demande une acquisition de dix valeurs, pour lesquelles une moyenne est faite, pour chaque coin de la carte ; ceci permet une homogénéisation des valeurs, pour éviter les bugs provenant de valeurs aberrantes. Nous avons donc à ce moment du code quatre valeurs de moyenne, appelées moyenneBD, moyenneBG, moyenneHD et moyenneHG (une pour chaque coin, les initiales correspondent à Bas pour B, Haut pour H, Droite pour D et Gauche pour G).

Étant donné que nous voulons déplacer notre tournesol dans les directions haut, bas, droite et gauche, nous devons donc créer les variables moyB, moyH, moyD et moyG qui prennent la moyenne de leurs deux variables associées (par exemple, pour moyB, la moyenne entre moyenneBD et moyenneBG).

Ensuite, afin de savoir dans quelles directions doivent se diriger les moteurs, deux tableaux sont utilisés. Un tableau permet de savoir si le déplacement ce fait vers le haut ou le bas (valeur de 'h' ou 'b'), l'autre permet de savoir si le déplacement se fait vers la gauche ou la droite (valeur de 'g' ou 'd').

Enfin, deux boucles permettent de faire bouger chacune un moteur. Par exemple, si le premier tableau contient un 'h', la variable posver décrémente, puis le moteur se déplace vers cette valeur de l'angle. Il en est de même pour le deuxième moteur associé au deuxième tableau et à la variable poshor.

Une des difficultés principales que nous avons eu lors de la réalisation de ce code fut les nominations des coins et des quatre directions. Nous avons dû dessiner sur les moteurs les quatre directions afin de ne pas s'emmêler dans la réalisation du code. D'ailleurs, l'utilisation de variables dans les coins afin de déplacer nos moteurs dans les directions haut, bas, droite et gauche fut difficile à coder car plus facilement confusant. Une autre difficulté fut l'utilisation des tableaux, car pour le langage Arduino, une syntaxe spéciale (existant aussi en C) que nous ne connaissions pas, est à respecter. Puis, l'évaluation de la variation des deux variables de position ne fut pas facile non plus ; en effet, nous avions la direction de mouvement de moteurs, mais les moteurs sont placés à l'envers sur le montage. Enfin, nous avons dû tester chaque butée du moteur après le montage final réalisé, butée représentée par des valeurs d'angle maximum ou minimum pour les variables posver et poshor.

a) *Déclarations*

```
#include <Servo.h>
```
Pour inclure les servomoteurs et leurs fonctionnalités prédéfinies dans la bibliothèque “Servo.h”.

```
Servo myservover, myservohor;
```
Nomination des servomoteurs dans le code.

```
int posver = 90
int poshor = 100;
```
Positions initiales des moteurs, donc du montage.
ver=vertical et hor=horizontal. Ce sont les axes de rotation.

```
float moyH=0;
float moyG=0;
float moyD=0;
float moyB=0;
float maxHB=0;
float maxGD=0;
```
Différents réels que l’on utilisera pour le calcul des moyennes.

```
char str1[1];
char str2[1];
```
Les deux tableaux que l’on utilisera pour savoir dans quelle direction se dirigera l’ensemble.

```
int lightPinBD = A1;
int lightPinHD = A2;
int lightPinBG = A3;
int lightPinHG = A4;
```
Ceci correspond aux branchements des capteurs sur la carte électronique. Par exemple, le capteur en bas à droite est relié au port A1 sur l’Arduino.

```
int ledPin=13;
```
Branchement vers la Led de l’Arduino, allumée lors du fonctionnement.

```
int lightPin = 0;
```
Définit une broche pour le photoresistor de l’Arduino.

b) *Réglages et liaisons Arduino/moteurs*

```
void setup()
{
```
Début de la boucle de réglage.

```
 Serial.begin(9600);
```
Initialisation de la communication série. Définit la vitesse de transmission (9600 bits/sec).

```
 pinMode( ledPin, OUTPUT );
```
Led définie en sortie.

```
 myservohor.attach(9);
 myservover.attach(8);
```
8 et 9 sont les numéros des ports de l’Arduino vers lesquels sont reliés les moteurs.

```
}
```

c) *Acquisitions luminosité*

```
void loop()
{
```
Début de la boucle principale.

```
 analogWrite(ledPin, analogRead(lightPin)/4);
```
Envoie d’un signal analogique à la Led.

```
 int moyenneBD =0;
 int moyenneBG =0;
 int moyenneHD =0;
 int moyenneHG =0;
```
Déclarations d’entiers dont nous aurons besoin.

```
 for (char i =0; i<=9;i+=1)
```
Boucle d’acquisition des 10 valeurs des capteurs.

```
{
        moyenneHG += analogRead(lightPinHG);
        moyenneBG += analogRead(lightPinBG);
        moyenneBD += analogRead(lightPinBD);
        moyenneHD += analogRead(lightPinHD);
        delay(10);
}
```

Pour chaque tour de boucle, la valeur récupérée est ajoutée aux précédentes.

Définit le délai de récupération des valeurs des capteurs à 10ms.

d) Calculs des moyennes

```
moyenneHG /= 10;
moyenneBG /= 10;
moyenneBD /= 10;
moyenneHD /= 10;
moyH=(moyenneHG+moyenneHD)/2;
moyD=(moyenneHD+moyenneBD)/2;
moyG=(moyenneHG+moyenneBG)/2;
moyB=(moyenneBG+moyenneBD)/2;
```

Finit le calcul des moyennes (théoriquement, ceci est inutile, mais en cas de dysfonctionnement d'un capteur, on connaîtra la valeur qu'il renvoie).

Calcul les moyennes des quatre directions.

e) Remplissage des tableaux

```
if (moyH>moyB)
{
        maxHB=moyH;
        str1[0]='h';
}
else
{
        maxHB=moyB;
        str1[0]='b';
}
```

Si le haut reçoit plus de lumière que le bas, on stocke 'h' dans le tableau correspondant.

Sinon on stocke 'b' dans ce même tableau.

```
if (moyG>moyD)
{
        maxGD=moyG;
        str2[0]='g';
}
else
{
        maxGD=moyD;
        str2[0]='d';
}
```

Si le côté gauche reçoit plus de lumière que le côté droit, on stocke 'g' dans le deuxième tableau.

Sinon on stocke 'd' dans ce même deuxième tableau.

f) Mouvements en conséquence

```
switch(str2[0])
{
case 'g':
```

Dans le cas où le deuxième tableau contient...

- 'g', alors :

```
        poshor --;                          1) on décrémente la variable poshor
        if  (poshor<55) poshor=55;          2) dans le cas où la butée est atteinte, on rectifie.
        myservohor.write(poshor);           3) on déplace le moteur à l'angle de valeur poshor.
        Serial.print("haut");               (pour identifier un éventuel problème).
        break;
case 'd':                                   - 'd', alors :
        poshor ++;                              1) on incrémente la variable poshor
        if (poshor>100) poshor=100;             2) idem
        myservohor.write(poshor);               3) idem
        Serial.print("bas");                    idem
        break;
default:
        Serial.println('erreur');           Dans le cas où le tableau ne contient ni 'g', ni 'd', pour
}                                           identifier un problème

switch(str1[0])                     Même fonctionnement que pour l'autre tableau, avec la
{                                           variable posver
case 'h':
        posver --;
        if (posver<0) posver=0;
        myservover.write(posver);
        Serial.println("gauche");
        break;
case 'b':
        posver ++;
        if (posver>180) posver=180;
        myservover.write(posver);
        Serial.println("droite");
        break;
default:
        Serial.println('erreur');
}
delay(10);                                  Fixe le délai à 10ms.
}
```

C. Tests finaux

Lorsque notre code fut complètement terminé, nous avons réalisé une série de tests sur notre montage.

Initialement, nous avions posé sur l'ensemble un gros panneau photovoltaïque, couvrant toute la carte électronique, puis ajouté deux petits panneaux sur les côtés. Cependant, il s'est avéré après les premiers tests que le gros panneau était trop lourd, et donc que les moteurs n'était pas assez puissants pour soulever et bouger l'ensemble. Nous avons donc retiré le gros panneau et laissé uniquement les deux petits sur la carte.

Lors de l'avant-dernière séance, le montage final était terminé. Il ne restait donc plus qu'à tester l'ensemble du montage. Le premier test fut concluant. En effet, notre tournesol électronique s'est parfaitement comporté dès le premier test. Nous avons même remarqué que même lorsque l'orientation n'est pas totalement optimale, notre tracker recherche et trouve la position idéale ; il se

réoriente toujours de façon à être en face de la lumière, perpendiculairement. Nous avons donc décidé de filmer ces mouvements lors de cette avant-dernière séance.

Cependant, nous avons remarqué que chacun des moteurs ne peut se déplacer que sur un demi-cercle de 180°, et non un tour complet. Il nous faudra donc, si on veut utiliser ce tracker, l'orienter de telle façon qu'il ne trouve pas de butée lors de sa rotation, c'est-à-dire que l'angle soit à 90° aux alentours de 13h (midi heure Solaire, en France nous sommes GMT+01), lorsque l'orientation du Soleil est plein Sud.

3) Montage de la carte électronique

Pour pouvoir mener à bien notre projet nous nous sommes dits que le meilleur moyen de fixer nos panneaux solaires serait de le faire directement sur la carte électronique. En effet, celle-ci présentait de nombreux avantages. Elle nous permettait d'enlever tous les fils en les remplaçant par des pistes. De même, tous les composants allaient être directement fixés sur cette carte et une fois le montage réalisé et fonctionnel, celui-ci allait rester définitif. De plus, avec cette carte il allait être plus aisé de fixer les capteurs solaires aux quatre coins du panneau photovoltaïque.

Pour pouvoir réaliser cette carte, il fallait préalablement la dessiner et en faire les plans. C'est donc à ce moment que le logiciel KiCad a été présenté à Baptistin. Ce logiciel devait permettre la réalisation de ce type de circuit. La prise en main n'a pas été très facile, il a fallu s'adapter et suive les différentes étapes afin d'obtenir un résultat probant.

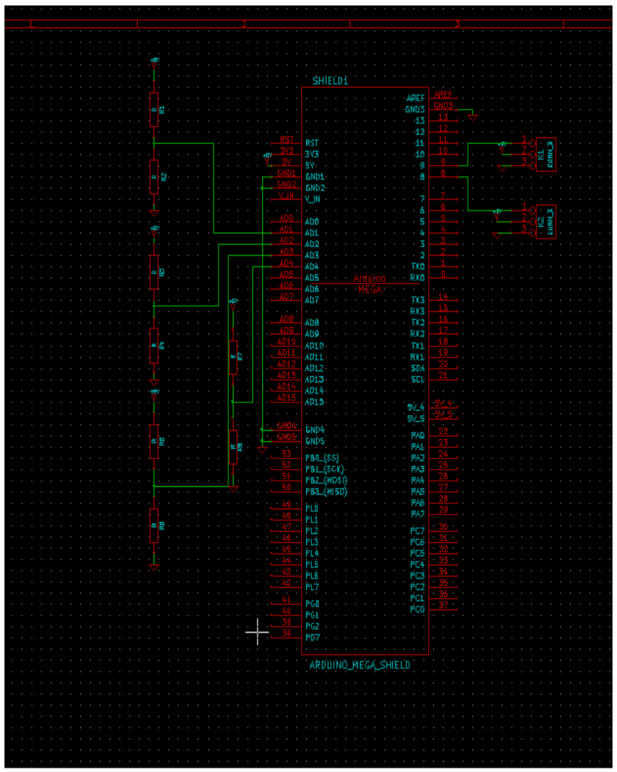
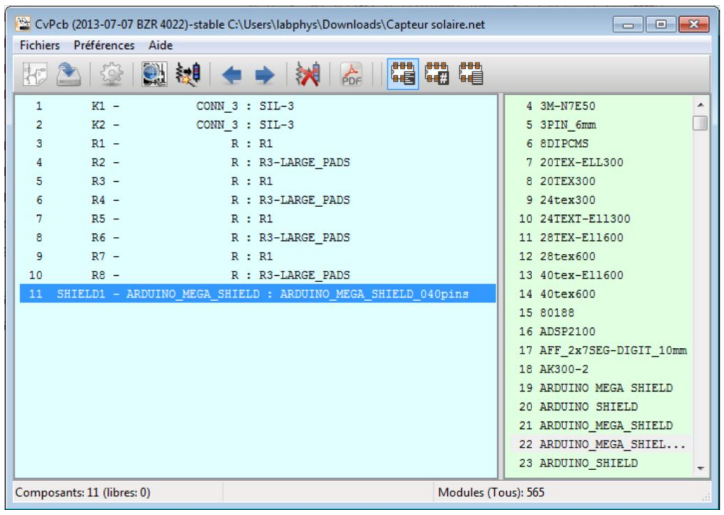


Figure 8. Fenêtre d'Eeschema

La première de ces étapes consistait à travailler avec Eeschema afin de choisir les différents composants que nous nécessitions pour ce montage, mais aussi leurs formes ou leurs caractéristiques comme la valeur de leurs résistances.

Malgré un nombre de composants impressionnant, il a parfois fallu importer des bibliothèques à partir de sites internet spécialisés pour pouvoir poursuivre notre travail. La carte Arduino Méga Schield en est un exemple. Quand tous les composants ont été rassemblés, il a fallu les rattacher avec des liaisons pour matérialiser les bornes d'entrées et de sorties. Finalement nous avons obtenu le schéma suivant.

Une fois ce premier schéma dessiné, nous avons dû poursuivre notre travail en créant une netlist avec le logiciel CvPcb. Ce fichier avait pour but d'associer tous les composants numérotés sur le schéma précédent à des composants réels dotés de différentes propriétés.



Cette netlist est indispensable pour le bon déroulement des opérations suivantes. De plus, cette netlist pouvait aussi nous permettre de modifier facilement la nature d'un composant ou sa caractéristique. Ce fichier permettait en fait d'associer les différents symboles créés sur Eeschema avec une empreinte réelle.

Figure 9. Fenêtre de la Netlist

Une fois cette étape passée, il nous a fallu travailler avec Pcbnew pour implémenter (répartir géographiquement), les composants pour que les capteurs solaires soient bien aux quatre coins du panneau solaire par exemple. Nous avons donc dû matérialiser le panneau solaire qui devait être placé à l'origine. C'est le carré blanc qui le matérialise sur le schéma suivant. Cependant nous avions une autre contrainte. En effet, la taille des cartes électroniques que nous pouvions imprimer était limitée. Par conséquent nous avons donc dû la matérialiser à son tour pour être certain que le montage soit réalisable. Les limites de la carte sont donc représentées en jaune sur la photo suivante.

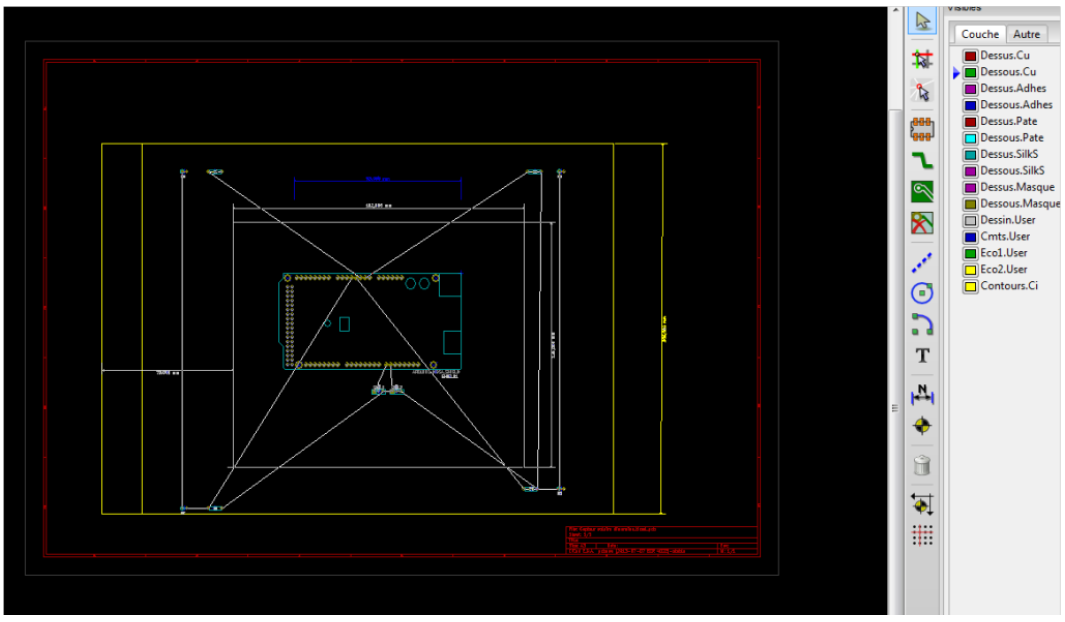


Figure 10. Limites de la carte

La dernière étape consistait à créer les liaisons remplaçant les fils qui allaient être imprimés en même temps que la carte. Bien que nous bénéficiions d'une aide de la part du logiciel qui indiquait quel composant devait être relié avec quelle pastille, la tache s'est révélée plutôt ardue. Effectivement, il a fallu faire très attention à ce que les fils ne se croisent pas pour ne pas avoir de court circuit. Cette tâche peut paraître simple au premier abord mais en réalité elle s'est avérée être un véritable casse-tête car nous manquions d'expérience dans ce domaine. Une fois les liaisons correctement établies, l'alimentation et la masse reliées aux composants leur correspondant, nous avions pratiquement terminé notre travail. Il ne nous restait plus qu'a vérifier que chaque angle n'était pas supérieur à 45° pour assurer une vitesse de réaction optimale de la part du circuit.

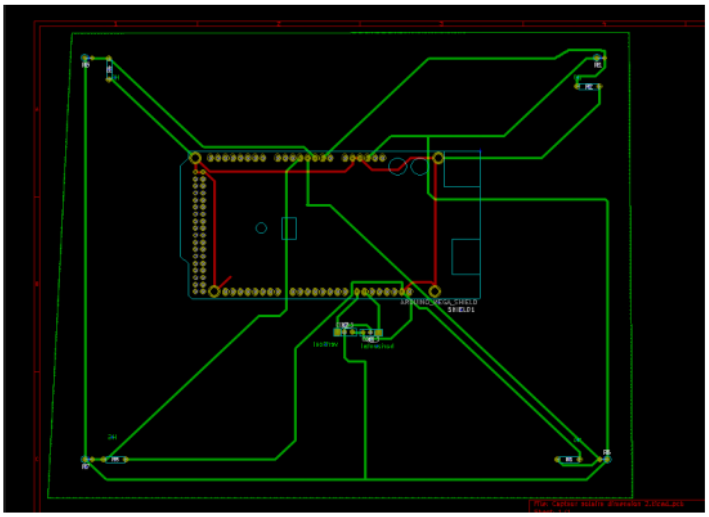


Figure 11. Création des liaisons à imprimer

Conclusion

Pour conclure, nous pouvons dire que nous avons réussi à réaliser le principal objectif de ce projet, soit monter et faire fonctionner un tournesol photovoltaïque, s'adaptant aux trajectoires de lumière afin d'en récupérer le maximum possible.

Toutefois, il nous semble important de signaler que nous n'avons pu réaliser la globalité de nos objectifs prévus, à cause d'une mauvaise estimation de notre temps. Si cela nous porte préjudice dans ce projet, nous pouvons en tirer les conséquences et un certain apprentissage pour les futurs projets à venir.

Des idées à développer sur ce même sujet pourraient être justement de s'intéresser en détail à la comparaison entre notre tournesol mobile et un panneau fixe. Le nôtre serait-il plus intéressant en rendement total malgré l'énergie dépensée par les servomoteurs ?

Enfin, en plus de sa fin technologique, ce projet nous a enseigné comment travailler en groupe, mettre en pratique la théorie, l'autonomie ou encore le développement personnel. Une expérience enrichissante, tant sur le plan humain que scientifique.

Sources

Introduction

http://jeunes.edf.com/article/le-solaire-photovoltaique-en-france,176

1ère partie

http://pybar.fr/index.php?page=tracker

http://fr.wikipedia.org/wiki/Cellule_photovolta%C3%AFque

http://fr.wikipedia.org/wiki/%C3%89nergie_solaire_photovolta%C3%AFque

http://www.observatoire-energie-photovoltaique.com/

http://www.photovoltaique.info/

http://blog.solorea.com/tracker-solaire

http://www.limpulsion.fr/art/FIT0045/DFROBOT__KIT_BRAS_ROBOT_5KG_4_8_6V_POUR_ARDUINO

3ème partie

http://www.limpulsion.fr/fab/ARDUINO

http://arduino.cc/en/Main/arduinoBoardMega2560

http://www.elektronique.fr/logiciels/kicad.php

www.kicad-pcb.org/download/.../eeschema.pdf

Table des illustrations