Compte rendu du projet arduino

Faucheux Aurélien et Faucheux Clément

31 mai 2018

1 Introduction

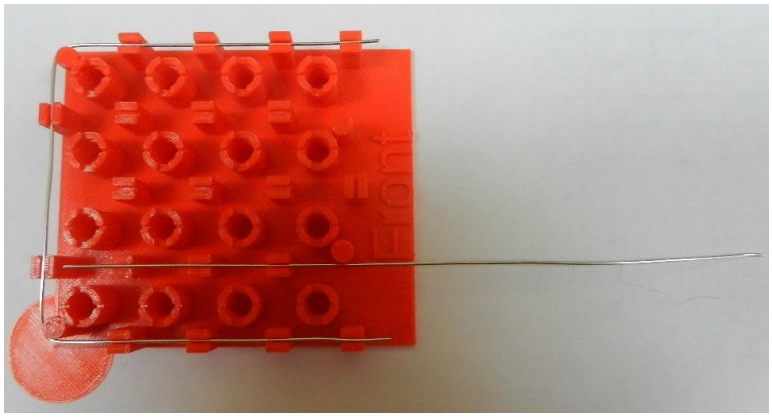
L'objectif final du projet a évolué au cours des premières semaines.
Au départ, nous avions décidé de réaliser un cube 4x4x4 avec des led bleues et qui fonctionnerait avec des multiplexeurs.
Nous avons ensuite découvert l'existence de led rgb programmable, répondant au nom de WS2812D-F8. Elles ne requièrent donc pas l'utilisation de multiplexeur. Nous avons alors décidé de réaliser un cube 5x5x5 avec ces led et dont les effets visuels seraient contrôlable à distance via une connexion bluetooth depuis notre téléphone.

2 Prise en main des led

Nous avons tout d'abord chercher à faire fonctionner ces nouvelles led qui comportent 4 pattes : VDD, GND, DIN qui récupère le signal d'entrée qui sera traité par le micro processeur intégré au led, et enfin DOUT qui transmet l'information à la led suivante. Ces led se branchent donc en « ruban » et ne nécessite l'utilisation que d'un seul pin de l'arduino.
Nous avons ensuite cherché une librairie capable de les faire fonctionner. Nous avons perdu beaucoup de temps à cette étape car nous avons trouvé des informations contradictoires sur internet, ce qui nous a fait douté de tout, de la carte arduino au fil reliant celle-ci au circuit, censé permettre l'allumage d'une led. En effet, on a d'abord pensé que la librairie utilisée (adafruit neopixel) ne supportait pas nos led. On a donc cherché d'autres librairies tels que FastLed ou d'autres crées par des internautes. Mais le problème ne venait pas des librairies. Le dysfonctionnement était dû au fait qu'il fallait ajouter une résistance avant l'entrée de la première led même si les led disposent de résistance intégrée.
On a ensuite apprivoiser la librairie Adafruit Neopixel pour réaliser quelques fonctions.

3 Modèle de construction du cube

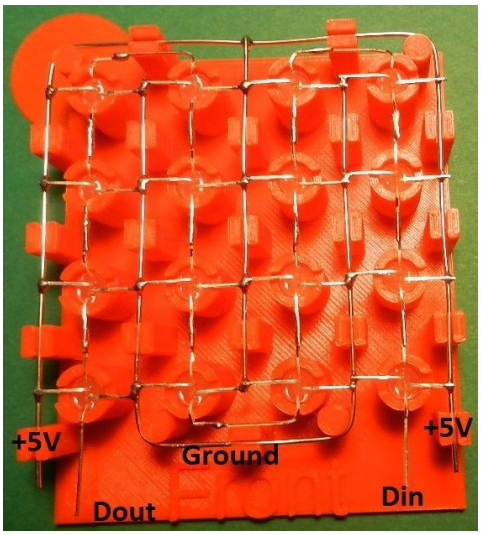
Nous avons tout d'abord eu l'idée de faire 25 colonnes de 5 led chacune qui positionnées côte à côte forme un cube. Chaque colonne étant constitué de deux tiges métallique une étant relié à vdd, l'autre à gnd. Nous avons abandonner cette idée car cela impliqué beaucoup de fils, de soudure pour un rendu moins esthétique qu'escompté (led ombragé).
Nous nous sommes donc inspiré d'une méthode qu'on a vu sur internet. Cette méthode consiste tout d'abord en la réalisation d'un gabarit servant à construire un étage bien droit et à faciliter les soudures. Voici le gabarit que l'on souhaite réaliser :



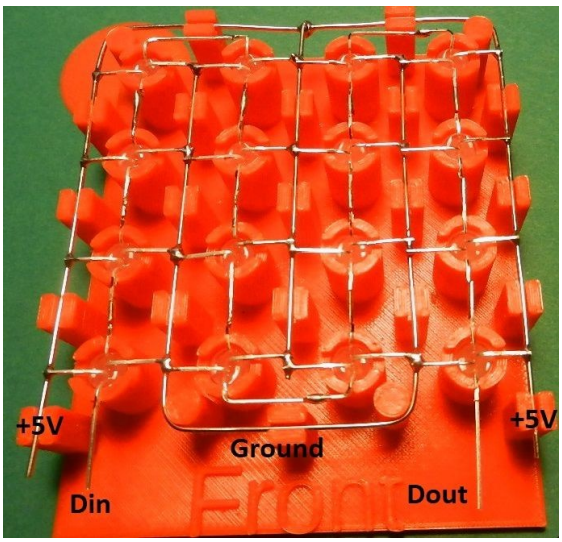
Comme on peut le voir ci-dessus, les cylindres accueillent les chapeaux de led, ce qui permet de faire les liaisons DIN/DOUT facilement en pliant les pattes des led dans les encoches prévu à cet effet. Les petits rectangles avec encoche ont pour rôle d'accueillir les « bus » VDD et GND de façon à ce que chaque led soit relié à ceux-ci. Cela présente également l'avantage de créer un quadrillage permettant à l'étage d'être bien solide, rigide.
On a ensuite remarqué que les étages pairs et impairs devaient se faire différemment. En effet, le DOUT finale d'un étage doit être relié au DIN initale de l'étage supérieur. S'il on fait tous les étages de la même manière, on va devoir faire cette liaison avec un fil qui va traverser toute la longueur du cube, ce qui n'est pas très esthétique. On doit donc inverser le sens de circulation entre deux étages successifs et, en conséquence, on doit inverser les positions des bus de façon à ce que, lorsqu'on superpose deux étages, les bus vdd sont tous les uns aux dessus des autres (de même pour gnd). Une autre raison de cette inversion de bus est qu'il faut également penser aux tiges qui vont relier verticalement les étages entre eux (pour les assembler). En effet, une tige qui va parcourir le cube en hauteur ne doit pas successivement être rattacher à vdd puis à gnd.

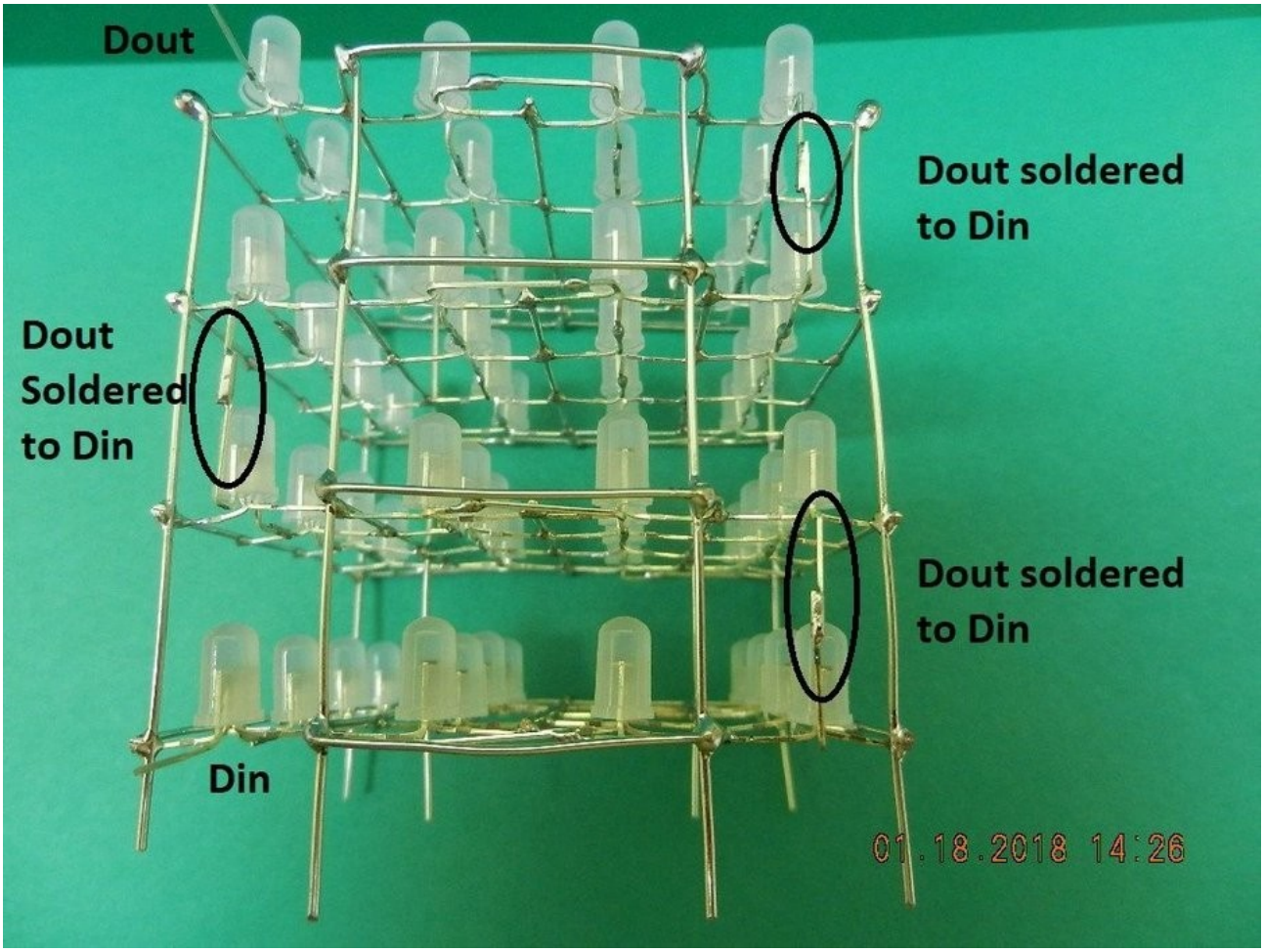
On a donc 4 tiges permettant d’assembler le cube (une dans chaque coin), 2 tiges qui seront uniquement relié aux bus vdd de chaque étages et de même pour gnd. On obtient donc ceci :

impaire



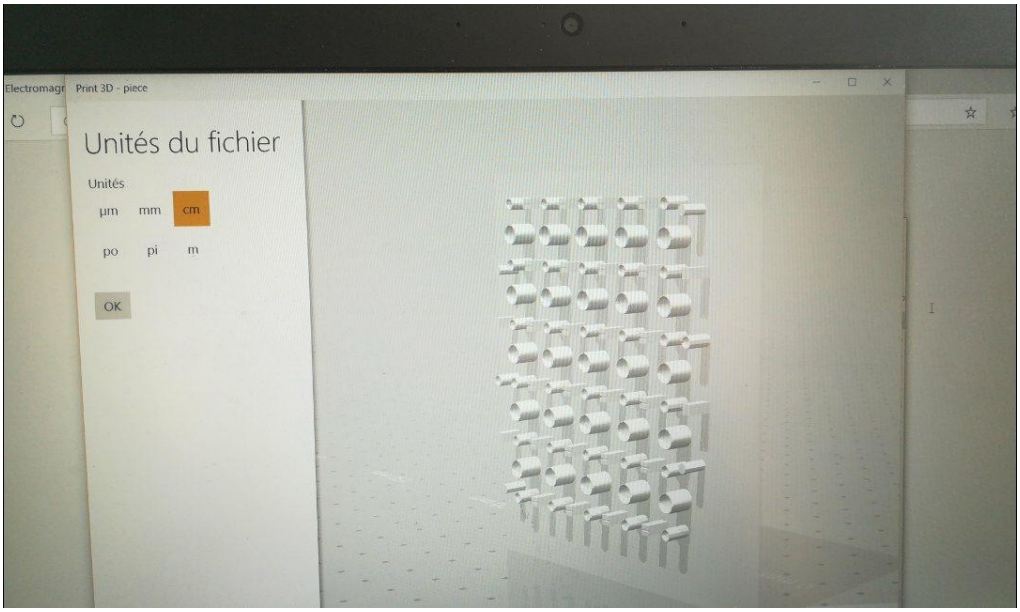
pair



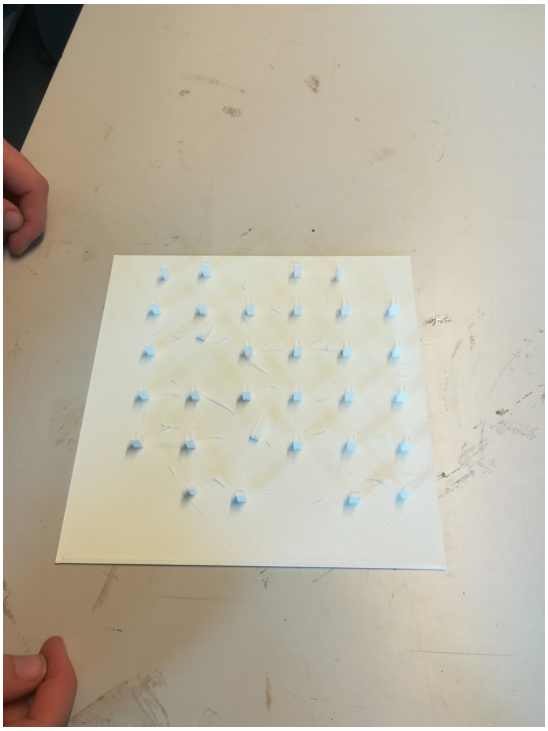


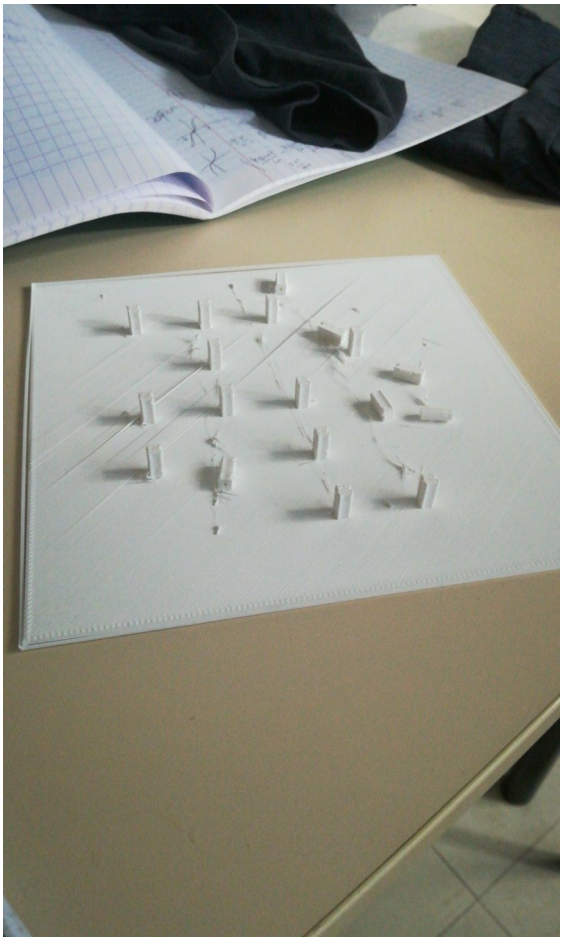
4 Construction

On a donc voulu reproduire ce gabarit adapté à notre taille de cube.
On l’a donc modélisé sur Autodesk :



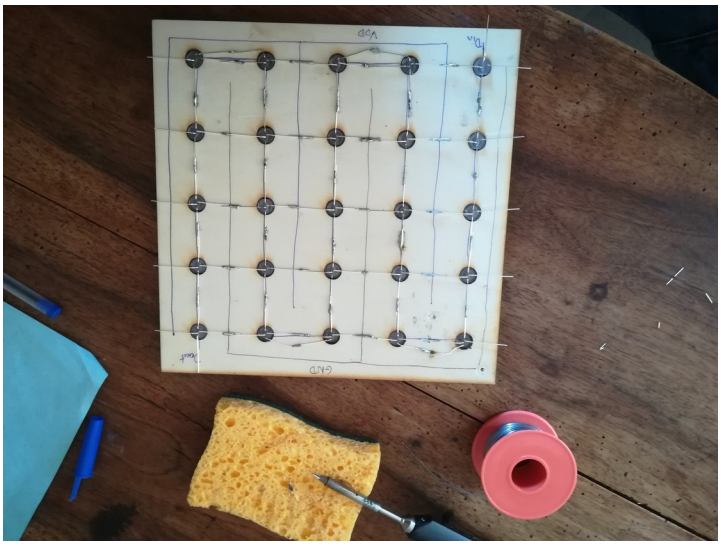
On s’est donc rendu au FabLab pour imprimer cette pièce mais le résultat n’a pas été celui attendu :

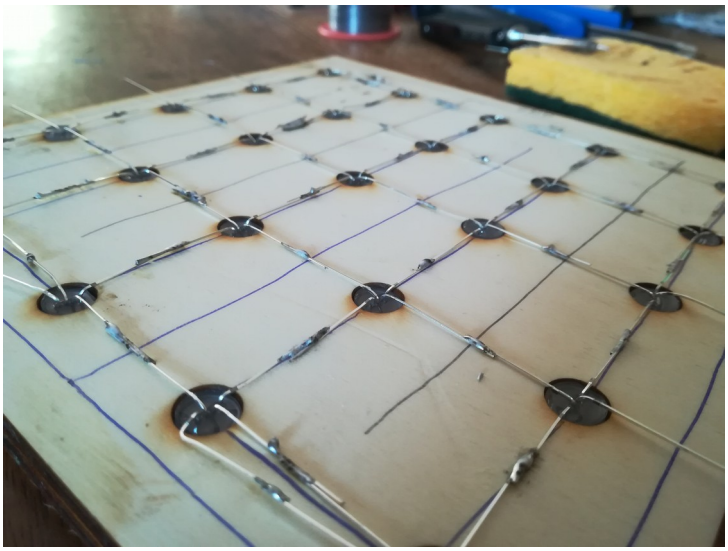




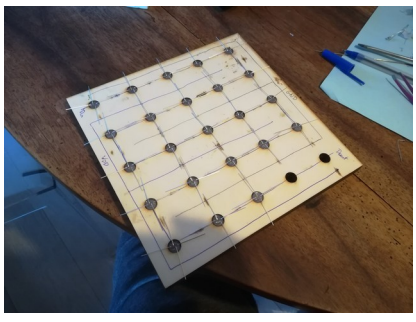
Aucun des cylindres ne s’est imprimé ! De plus, le peu qui s’est imprimé ne tient debout que par un fil…

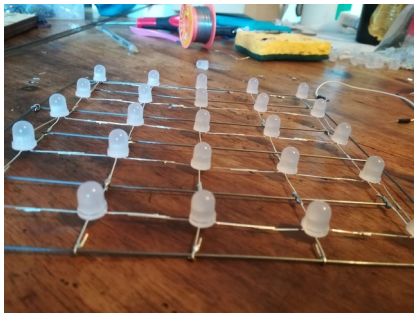
On alors changé de support tout en gardant l’idée du gabarit, des bus, etc. On a choisi de percer des trous de 11mm (hauteur du chapeau d’une led) de profondeur dans une planche de bois pour imiter les cylindre du gabarit de départ. Ne disposant que de planches de 6mm d’épaisseur nous réalisons les trous à l’aide de la découpeuse laser sur deux planches que nous avons collé ensuite. Nous avons choisi d’espacer les led (de centre à centre) de 39,2mm car nous voulions 3cm entre des chapeaux de led. Nous nous sommes procuré en conséquence des tiges de fer pour les bus. Nous obtenons donc ceci :

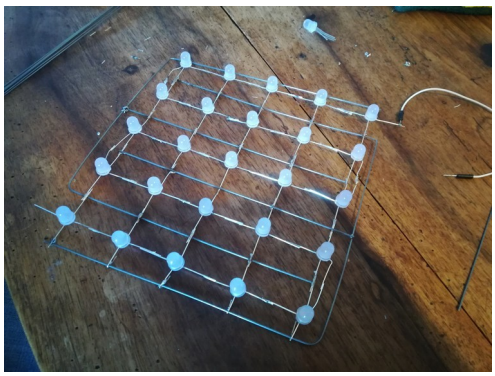




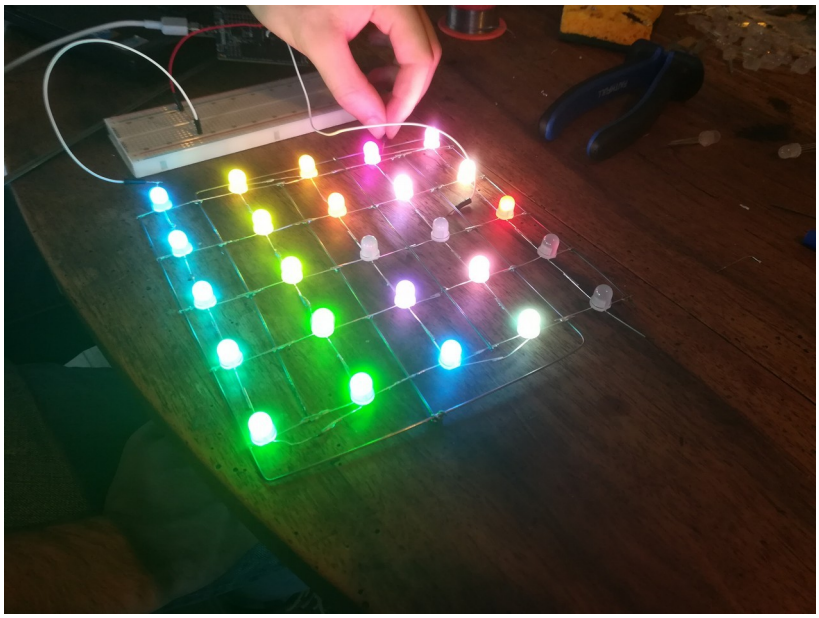
Nous réalisons donc nos 5 étages à l’aide de notre gabarit :





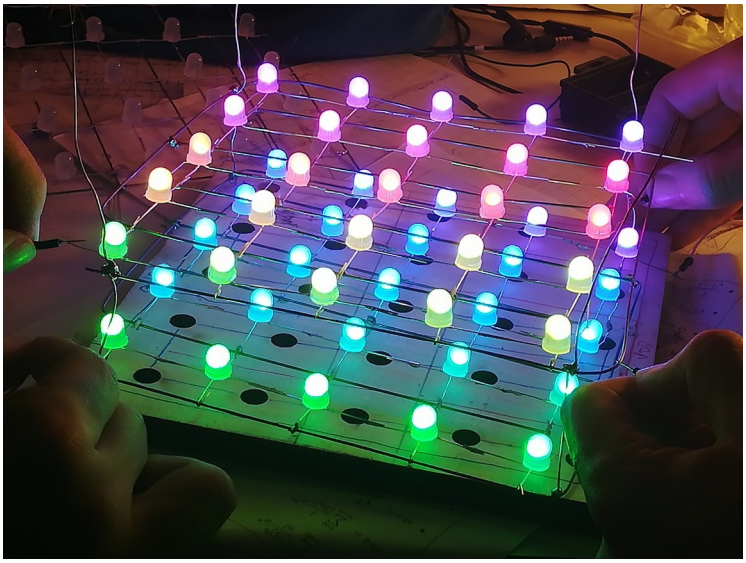


Nous testons ensuite chaque étage de façon indépendante. Après quelques problèmes mineurs tels que des soudures mal faites ou une led qui explose, nous parvenons à allumer correctement 3 étages parmi 5. Les 2 plans autres s'allument mais de manières aléatoire, c'est à dire que des led au hasard s'allument puis s'éteignent très rapidement. De plus, déplacer vdd et gnd perturbe l'allumage de ces plans et même parfois des 3 plans fonctionnels. Parmi les étages fonctionnels, parfois certaines led s'allument d'une couleur différente que celle voulue. On a remarqué aussi que, parmi les non fonctionnels, on a souvent observé le cas où seul les 9 premières led s'allument correctement tandis que les 16 autres restent éteintes.
On obtient alors des résultats comme celui ci :

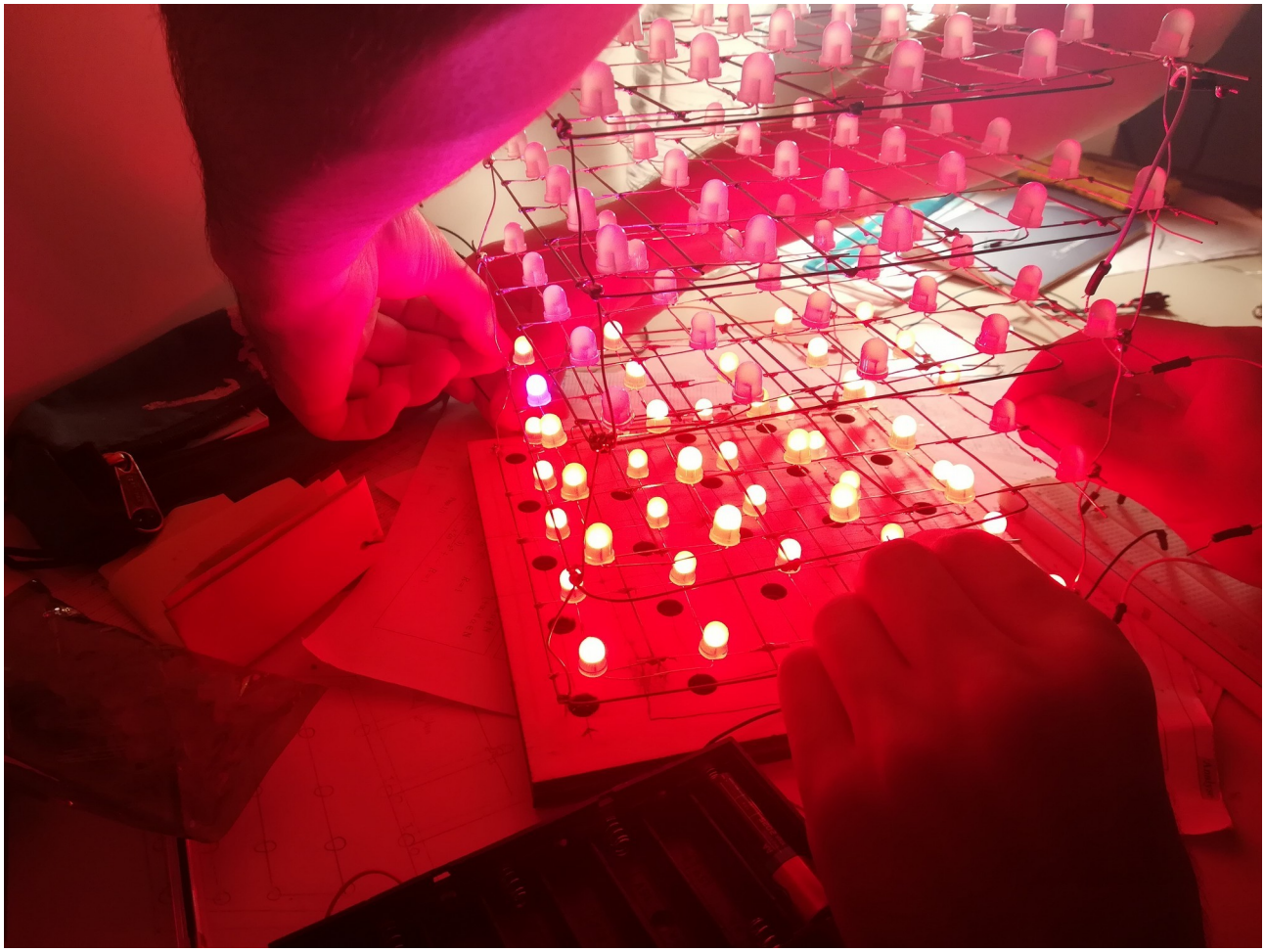


Nous refaisons alors quelques soudures que nous estimons mal faites et qui seraient cause de mauvaise liaison led/led ou led/bus ou bus/bus et nous remplaçons des led que nous accusons de ne pas fonctionner du tout.
Le résultat est décevant puisque aucun changement remarquable n'est observé.
Nous soupçonnons alors la basse qualité de l'étain utilisé pour les soudures ainsi que les tiges de fer qui ne sont pas faites pour la soudure puisque l'étain fondue glisse sur ces tiges comme des gouttes d'eau sur une vitre.
Nous décidons alors de changer de fil de fer pour assembler des étages.

Nous assemblons alors le cube en commençant par les 3 étages fonctionnels. Nous réalisons un test avec 1 puis 2 étages avant d'ajouter les 3 étages suivant :



Le résultat est concluant. Avec 5 étages, les 2 premiers restent fonctionnels, le 3e, qui fonctionnait en indépendant, ne fonctionne que partiellement. Les 2 derniers étages ne s'allument presque jamais ou avec des « bugs» d'allumages décrit précédemment. On obtient le résultat suivant :



5 Code

Voici quelques lignes de code utilisées pour créer des pattern sur un plan de led. Étant donné l'impossibilité de tester le cube dans sa totalité, il n'y pas de code effectuant des effets visuels 3D

test-led.ino

```
#include <Adafruit_NeoPixel.h>

#define PIN 6

Adafruit_NeoPixel cube = Adafruit_NeoPixel(10, PIN, NEO_RGB + NEO_KHZ800);


void setup() {
  // put your setup code here, to run once:
  cube.begin();
  cube.show();
  cube.setBrightness(255);



}

void loop() {
  //  put your main code here, to run repeatedly:
  flag(250);
  shut(1000);
  aleatoire(10);
  shut(5000);
  //switchOn(cube.Color(255, 0, 0));
  brightnessDown(30);
  switchOn(cube.Color(0, 255, 0));
  brightnessUp(10);
  brightnessDown(30);
  switchOn(cube.Color(0, 0, 255));
  brightnessUp(20);
  brightnessDown(20);
  shut(1000);
  cube.setBrightness(255);
  allerRetour(cube.Color(100, 100, 255));
```

```
}
void flag(int att) {                // drapeau français !
  int i = 0;
  for ( i = 0; i < cube.numPixels(); i++) {
    if (i % 3 == 0) cube.setPixelColor(i, 0, 0, 255);
    if (i % 3 == 1) cube.setPixelColor(i, 255, 255, 255);
    if (i % 3 == 2) cube.setPixelColor(i, 255, 0, 0);
    //cube.setBrightness(64);
    cube.show();
    delay(att);


  }
}

void shut(int att) {                  //éteind tout le ruban pendant un certain temps
  for (int i = 0; i < cube.numPixels(); i++) {
    cube.setPixelColor(i, 0, 0, 0);
  }
  cube.show();
  delay(att);
}
void shut() {                  //éteind tout le ruban
  for (int i = 0; i < cube.numPixels(); i++) {
    cube.setPixelColor(i, 0, 0, 0);
  }
  cube.show();
}
void aleatoire(int att) {             //allume un led aléatoire d'une couleur aléatoire pedant un temps aléatoire
  for (int i = 0 ; i < att ; i++) {
    cube.setPixelColor(random(0, cube.numPixels()), random(0, 255), random(0, 255), random(0, 255));
    cube.show();
    delay(random(100, 600));
    shut(300);
  }
  //delay(random(10,1000));
}



void brightnessDown(int v) {           //plus v est faible plus la baisse d'intensité est rapide
  for (uint16_t i = 255; i >= 1; i--) { // faut
    cube.setBrightness(i);
    cube.show();
    delay(v);
  }
}
void brightnessUp(int v) {
  for (uint16_t i = 1; i <= 255; i++) {
    cube.setBrightness(i);
    cube.show();
    delay(v);
  }
}
void allerRetour(uint32_t c) {           //allume une par une les leds
  for (int i = 0; i < cube.numPixels(); i++) {
    cube.setPixelColor(i, c);
    cube.show();
    delay(500);
    shut();
    if (i == cube.numPixels() - 1) {
      for (int i = cube.numPixels() - 2; i >= 1; i--) {
        cube.setPixelColor(i, c);
        cube.show();
        delay(500);
        shut();
      }
    }
  }
}
void switchOn(uint32_t c) {          //allume toutes led d'une couleur choisie
  for (uint16_t i = 0; i < cube.numPixels(); i++) {
    cube.setPixelColor(i, c);
    cube.show();
  }
}
```

```
Téléversement terminé
Le croquis utilise 4060 octets (13%) de l'espace de stockage de programmes. Le maximum est de 30720
Les variables globales utilisent 44 octets (2%) de mémoire dynamique, ce qui laisse 2004 octets pou
```

5 Conclusion

Nous ne sommes donc pas parvenu à allumer l'intégralité du cube de façon stable, ce qui nous empêche de satisfaire les objectifs annoncés au départ. En effet, nous ne pouvons pas créer une grande variété d'effets visuels en 3D, ni les contrôler via notre téléphone.
Nous aurions quand même pu réaliser la connexion Bluetooth mais celle ci n'aurait que peu d'intérêt étant donné le dysfonctionnement du cube. Nous avons préféré tenter de réparer le cube mais en vain.
Malgré tout, nous avons apprécié réaliser ce projet qui nous appris à gérer des problèmes de manière autonome.
Nous remercions M.Masson et M.Ferrero pour nous avoir permis de réaliser ce projet.