



# STROBOGLOVE

## **LES GANTS QUI MANNIPULENT LE TEMPS**

COMBAL Quentin, COCOGNE Romain

PEIP2 Polytech'Nice

Encadré par Mr Masson et Mr Ferrero



École d'ingénieurs



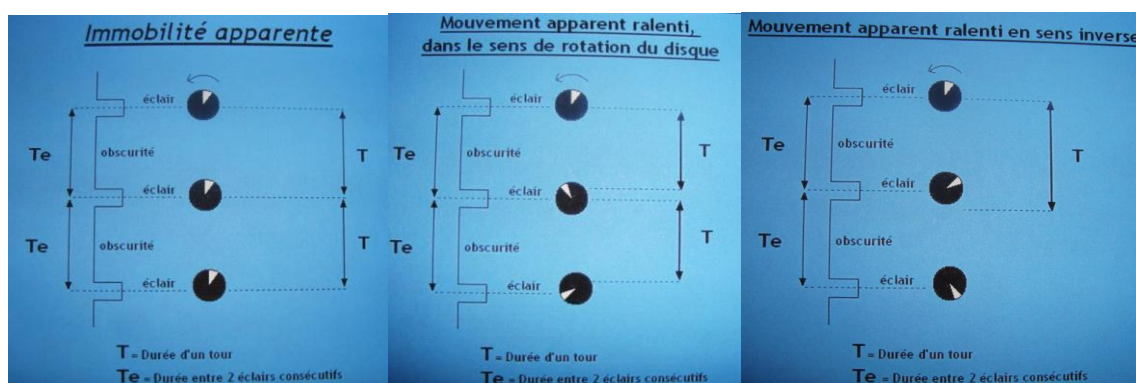
# SOMMAIRE

<b>INTRODUCTION .....</b>	<b>3</b>
FRANÇAIS.....	3
ENGLISH .....	3
<b>OBJECTIFS ET FONCTIONNEMENT.....</b>	<b>4</b>
ORGANISATION.....	4
<b>MATERIEL.....</b>	<b>6</b>
SCHEMAS.....	6
CARACTERISTIQUES.....	7
<i>AUTONOMIE.....</i>	<i>7</i>
<i>REGIME DE FONCTIONNEMENT.....</i>	<i>7</i>
<b>PROGRAMME .....</b>	<b>8</b>
MAITRE .....	8
ESCLAVE .....	10
<b>DIFFICULTEES .....</b>	<b>11</b>
<b>PERSPECTIVES .....</b>	<b>12</b>
<b>CONCLUSION .....</b>	<b>13</b>
CE QUE CE PROJET NOUS A APPORTE .....	13
REMERCIEMENTS .....	13
<b>SOURCES .....</b>	<b>14</b>

## FRANÇAIS

Notre projet est le Stroboglove. Le nom vient du mélange entre Stroboscopique et Glove, qui veut dire gant en Anglais. En effet notre projet consiste à utiliser l'effet stroboscopique de façon ludique et spectaculaire. Pour cela, nous utilisons une paire de gants, l'un émettant de la lumière pour éclairer un objet en rotation et créer l'effet stroboscopique, et l'autre contrôlant l'effet en inclinant la main de droite à gauche. Cela donnera l'illusion de contrôler l'objet en rotation juste avec un mouvement de la main.

Pour ceux qui ne savent pas ce qu'est l'effet stroboscopique, il s'agit d'une illusion d'optique un peu particulière, puisqu'elle ne dépend pas de l'œil humain pour fonctionner (ce qui peut donc servir d'appareil de mesure scientifique). En effet, le principe réside dans le fait d'éclairer par intermittence un objet en mouvement périodique. Si la fréquence de clignotement correspond à la fréquence du mouvement, l'observateur aura l'impression de voir un objet immobile.



Nous avons eu l'idée de ce projet en regardant le film *Dr Strange*, et plus particulièrement la scène où le personnage principal contrôle le temps d'un simple mouvement de main. L'inspiration nous a frappé, et nous nous sommes mis à imaginer les solutions pour recréer cet effet en vrai.

Pour la suite de ce rapport, nous allons vous aider à comprendre toutes les facettes de ce projet pour qu'au final vous puissiez vous-même créer vos propres Stroboglove.

## ENGLISH

Our project is called *The Stroboglove*. The name comes from the words "Stroboscopic" and "glove". Indeed, our project is about using the stroboscopic effect in a ludic and spectacular way. To do that, we are using a set of two gloves. The first one emits flashes of light to create the stroboscopic effect on a rotating object, and the other one controls the effect with the angle of the hand. This gives the illusion of controlling a rotating object with hand motion.

For those who are not familiar with the stroboscopic effect, it is an optical illusion which relies on flashing short pulses of light on a rotating object at regular intervals. If an object has a constant period of rotation, then flashing it with the same period between flashes will make the object seem motionless, while it is actually moving. This gives the impression of stopping time!

It is a particular optical illusion because it doesn't rely on the human eye. Thus, it can be used for scientific measurements.

The idea for this project came to us when we watched the *Doctor Strange* movie, more specifically the scene where the main character controls time with a simple hand gesture. We began searching ways to recreate the same effect in real life.

In this report, we will help you understand all the aspects of this project, so that in the end you could create your own *Strobogloves*.



L’objectif de notre projet est de concevoir un objet ludique et impressionnant. Nous voulions donc que le produit final soit à la fois ergonomique et design.

Nous avons choisi d’utiliser une paire de gants dans cet optique d’ergonomie. En effet l’utilisation en est plus intuitive et plus amusante qu’avec un simple potentiomètre par exemple.

Comme il y aura deux gants, un qui éclaire l’objet en rotation et l’autre qui commande, il faut établir une communication entre eux. Nous avons choisi le Bluetooth car c’était la solution la plus adapté à la situation.

Il fallait aussi que le produit soit adaptable à tout type d’objet en rotation, avec des vitesses pouvant être très différentes. Nous avons donc décidé d’implémenter deux modes, un mode de réglage macroscopique de la fréquence et un mode microscopique qui relève plus de l’ajustement. Pour passer d’un mode à l’autre nous avons imaginé que l’utilisateur aura juste à plier ses doigts en accordance du mode voulu. Pour générer cet effet nous avons pensé utiliser des résistances sensibles à la flexion.

Finalement, nous avons décidé de rajouter des LED indiquant à l’utilisateur si les deux gants sont connectés ou non.

## ORGANISATION

Nous avons prévu de suivre ce diagramme.

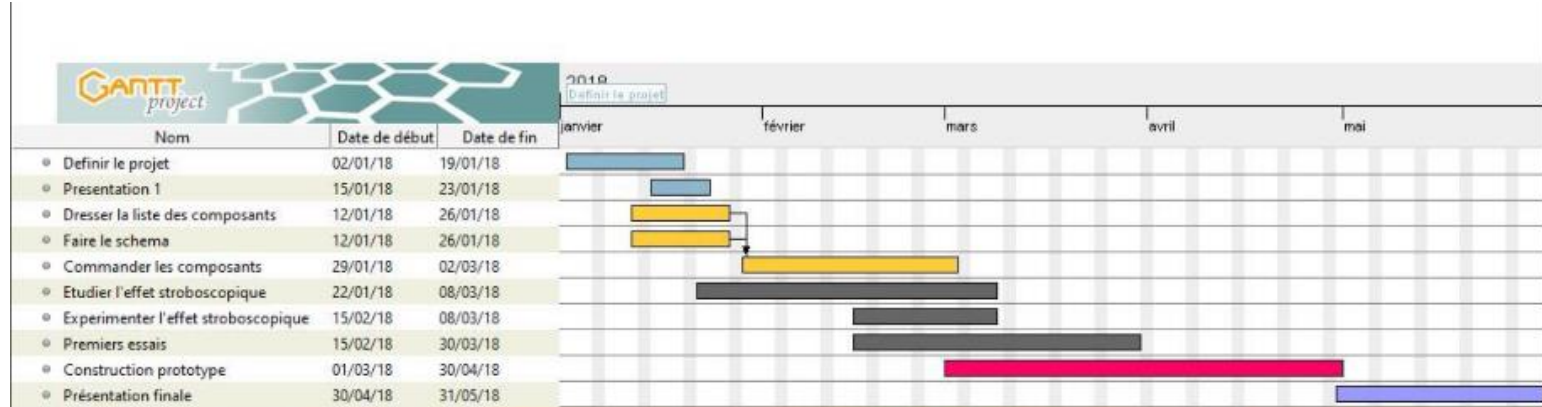


Figure 2 gantt project avant

Finalement voici comment s’est déroulé le projet.

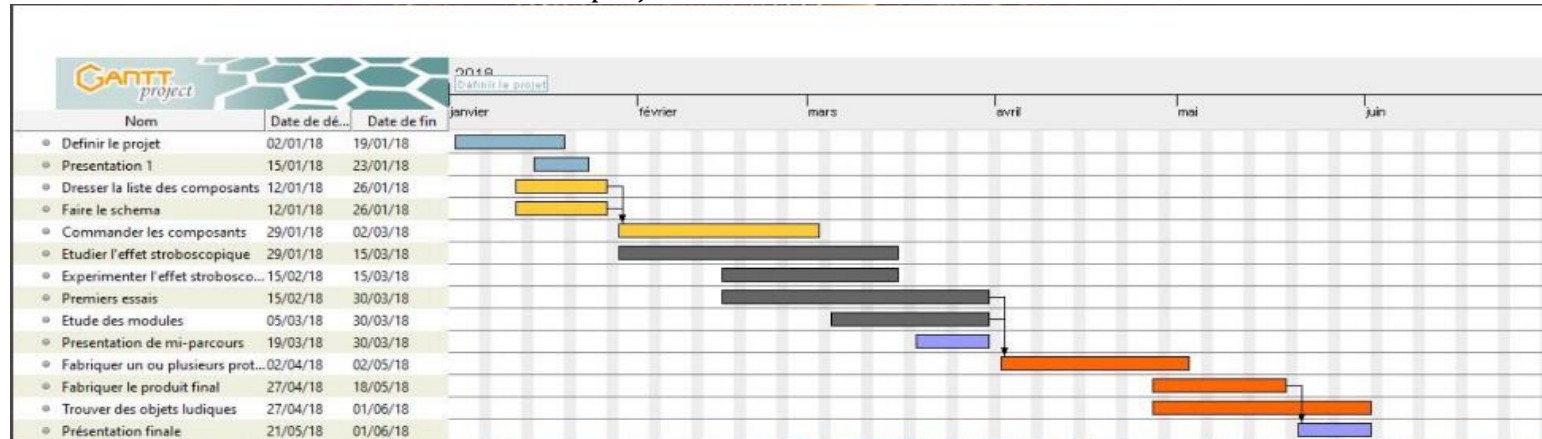


Figure 3 gantt project après

Notre Diagramme de Gantt c'est vu modifié au fur et à mesure. Nous avons notamment consacré beaucoup plus de temps que prévu à l'étude des modules et à la construction finale ; Cette dernière étape fut retardée à cause du délai de livraison des composants.

Malgré tout, nous avons plutôt bien respecté les dead lines que l'on s'était imposées, ce qui nous a permis de nous laisser de la marge pour pallier les imprévus.

Finalement, toutes les fonctionnalités prévues sont présentes sur la paire de gants. Nous avons même ajouté une fonctionnalité qui active/désactive la LED avec une secousse de la main.

Il reste cependant des problèmes de stabilité, il arrive de devoir éteindre puis allumer le gant maître plusieurs fois pour que la connexion s'effectue.

Il y a autre chose qui n'est pas vraiment respecté. Il s'agit du design. Celui-ci n'est pas parfait et les gants ne rendent pas aussi propre que prévu. Nous avons aussi dû rajouter une poche qui s'attache au bras pour contenir la batterie, le DCDC et l'Arduino et ne pas trop encombrer le gant.



*Figure 4 rendu visuel des gants*

MATERIEL

ITEM	PRIX en €
<b>Structure</b>	
Gants	Non payé
Brassard	4.90
Pâte thermique	5.24
Lentilles optiques	2.82
Plaque de cuivre	Non payé
<b>Electronique</b>	
2 Arduino (Nano/Mini Pro)	2x4.39
Accéléromètre	Non payé
Résistances 100 et 10k	Non payé
MOSFET	0.93
Flex Sensors	2x6.82
LED blanche 100W	6.83
<b>Alimentation</b>	
Batterie 11,1V - 12V +3A 1500 mAh	Non payé
Batterie 8V	Non payé
Convertisseur Boost DCDC (12V to 34V)	2.79
<b>Transfert de données :</b>	
Bluetooth HM-10	Non payé
<b>Objet en rotation</b>	
Moteur continu ou pas à pas	Non payé
Horloge	Non payé
TOTAL	45.93

Le matériel non payé peut être un matériel de récupération ou bien fourni par Polytech’Nice.

SCHEMAS

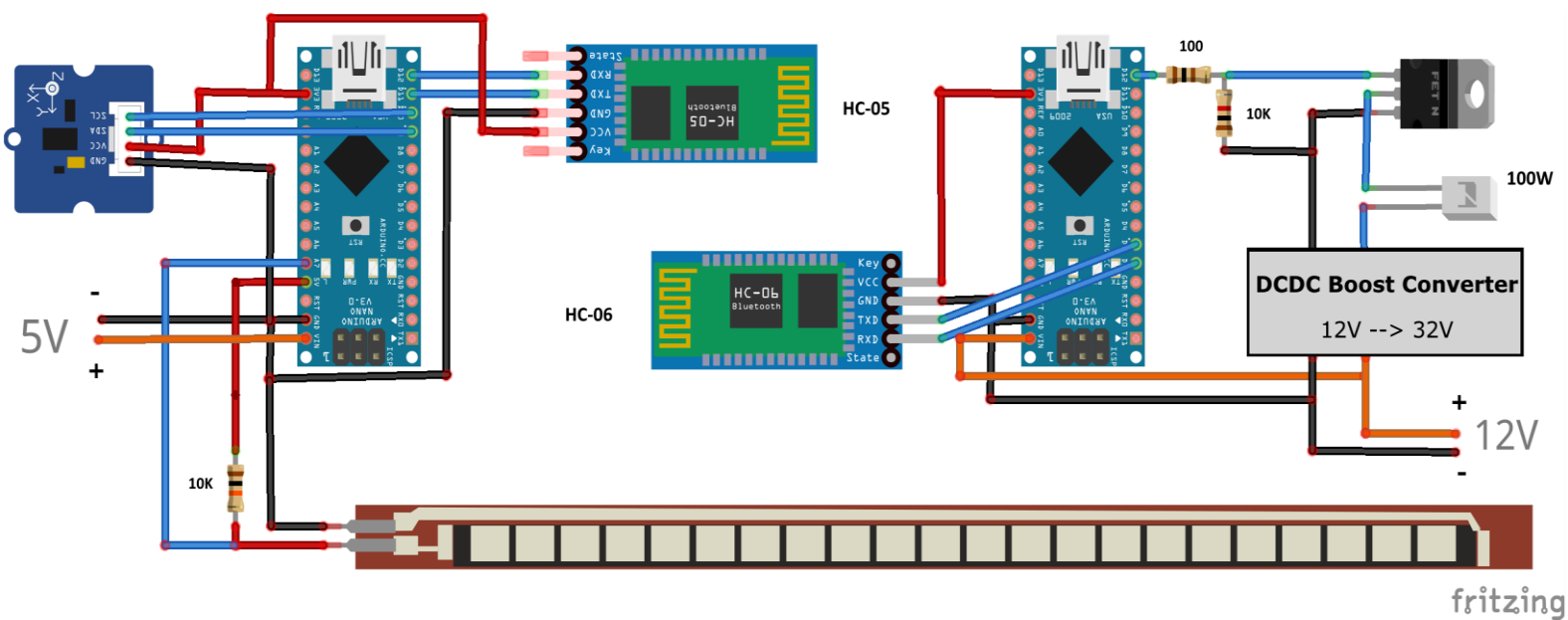


Figure 5 Schémas de câblage

*La partie gauche* est la partie maître. Elle comprend :

- Un accéléromètre qui détecte les inclinaisons de la main
- Deux flex sensors qui détectent si les doigts de la main sont pliés. Ceux-ci permettent de choisir le mode de fonctionnement du gant.
- Un module Bluetooth HM-10 pour envoyer la fréquence à l'esclave.
- Une Arduino nano pour coordonner les éléments : choisir le bon mode, traiter les données de l'accéléromètre, en déduire la fréquence de clignotement de la LED correspondante et la transmettre au module Bluetooth.

*La partie droite* est la partie esclave. Elle comprend :

- Un DCDC boost converter qui va transformer le 12V de la batterie en 32V pour que la luminosité de la LED soit suffisante.
- Une LED 100W. Il s'agit de l'élément clé du projet.
- Un transistor MOSFET avec des résistances pour faire clignoter rapidement la LED.
- Un module Bluetooth HM-10 pour recevoir la fréquence de clignotement de la LED.
- Une Arduino nano pour récupérer la fréquence, la transformer en un temp et faire clignoter la LED.

Nous avons par la suite décidé de rajouter des LED de couleur pour améliorer l'esthétisme et l'ergonomie du produit (non représentées sur le schéma par soucis de clarté).

## CARACTERISTIQUES

### AUTONOMIE

Consommation du circuit esclave : 1.4A en régime de fonctionnement classique ; 22mA LED éteinte.

Consommation du circuit maître : 50mA.

Batterie esclave : 1.3Ah.

Batterie maître : 200mAh.

Donc le gant maître possède 4h d'autonomie et le gant esclave 1h.

### REGIME DE FONCTIONNEMENT

On a déterminé les fréquences minimum et maximum de fonctionnement de la LED.

La fréquence minimum est déterminée par l'œil humain : si le clignotement est détecté à l'œil nu, la fréquence est trop faible.

La fréquence maximum est déterminée par les caractéristiques de l'Arduino. En effet le baud rate du Bluetooth limite la vitesse de clignotement.

Nous avons trouvé :  $F_{min}=60\text{Hz}$ ,  $F_{max}=500\text{Hz}$ .

# PROGRAMME

## Explication du programme Arduino.

### MAITRE

Le code de l'Arduino maître se décompose en trois parties :

- Le traitement des données de l'accéléromètre et des Flex Sensors:  
Le programme maître récupère les données brutes de l'accéléromètre, en sélectionnant celles qui sont utiles. Le programme effectue ensuite un calcul brut de l'angle de la main avec les valeurs *accX* et *accY*. En se servant de la vitesse angulaire de la main (*GyroZ*) et d'un filtre de Kalman, le programme calcule l'angle « filtré » de la main, ce qui permet de lisser le mouvement et éviter les sauts de valeur dus au bruit. Le programme ajuste ensuite les valeurs finales de délai (celles à envoyer au deuxième gant) en fonction de l'angle de la main.

```
void mesure_angle() {
    unsigned long currentMicros = micros();
    i2cRead(0x3B); //acquisition des données, demande à l'acceleromètre d'envoyer 14 entrées |
    accX = ((data[0] << 8) | data[1]);
    accY = ((data[2] << 8) | data[3]);
    accZ = ((data[4] << 8) | data[5]);
    gyroZ = ((data[12] << 8) | data[13]);

    //Calcul de l'angle avec la valeur de g sur z , la vitesse angulaire de z, et le filtre de Kalman
    accZangle = (atan2(accX, accY) + PI) * RAD_TO_DEG;
    gyroZrate = -((double)gyroZ / 131.0);
    gyroZangle += kalmanZ.getRate() * ((double)(currentMicros - timer_acc) / 1000000); // Calculate gyro angle using the unbiased rate
    kalAngleZ = kalmanZ.getAngle(accZangle, gyroZrate, (double)(currentMicros - timer_acc) / 1000000); // Calculate the angle using a Kalman filter
    timer_acc = currentMicros;
}
```

Le programme possède deux modes d'ajustement des valeurs : un mode macroscopique qui permet de faire varier la fréquence de ~60Hz à ~500Hz, et un mode microscopique qui permet un réglage plus minutieux (de l'ordre du Hertz). En fonction des valeurs de résistance des Flex Sensors, le programme ajuste la valeur du délai de façon macroscopique ou microscopique. L'utilisateur peut donc changer de mode en pliant certains doigts de la main.

```
flex_0 = analogRead(0);
flex_1 = analogRead(1);
/*Si au moins un des deux flex sensors sont pliés --> Reglage macro*/
if ((flex_0 > 960) || (flex_1 > 900)){ /*La barre du flex_1 est plus haute que l'autre car c'est celui qui est un peu plié par défaut */
    t_macro = map(kalAngleZ-last_angle , -120+last_angle , 120+last_angle , t_macro_min,t_macro_max);
    //Serial.print(F("Mode : macro ; "));
    couleur[0]=3; couleur[1]=1; couleur[2]=5;
}

/* Sinon --> Reglage micro */
else {
    t_micro = map(kalAngleZ,-120,120,t_micro_min,t_micro_max);
    last_angle = kalAngleZ;
    //Serial.print(F("Mode : micro ; "));
    couleur[0]=1; couleur[1]=5; couleur[2]=1;
}
```

- Le transfert des données via communication Bluetooth :  
On utilise la librairie SoftwareSerial. Le programme vérifie constamment l'état de la connexion Bluetooth en envoyant un '\$'. Si après l'avoir envoyé, il en reçoit un, alors il considère qu'il est connecté. Sinon, il essaye de se connecter au module.  
Après s'être connecté, il va envoyer à intervalle régulier la fréquence de clignotement, calculé dans la partie accéléromètre. Le message est précédé d'un '&' et suivi d'un '\n' pour rendre la connexion plus stable et éviter les erreurs.



```

char receive;          //sauvegarde le char reçus
if (BTSerial.available()){
    receive=BTSerial.read();
}
unsigned long currentMillis = millis();
if (currentMillis - timer_send_donnee >= time_send_data) {
    timer_send_donnee = currentMillis;
    if(t_micro+t_macro !=0){
        BTSerial.print("&");
        BTSerial.print(t_micro+t_macro);
        BTSerial.print("\n");
    }
}
test_connection("send",'$',' '); //test si on est connecté
test_connection("receive",'$',receive); //test si on est connecté

if(!is_connected){ //si on est pas connecté
    try_connection(); //on essaye
    chenillard(); //petit chenillard
}
else{ //sinon
    low_blink(); //on allume tout
}

```

- La strip LEDs :

Le programme utilise la library Adafruit\_NeoPixel pour contrôler les strip LEDs qui se trouvent sur le gant maître au niveau du poignet. Les strip LEDs changent de couleur et de mode de clignotement en fonction du mode choisi (macro ou micro) et de l'état du programme.

- Couleur des LEDs :
  - Violet : Mode Micro
  - Vert : Mode Macro
- Mode :
  - Clignotement doux : Indique que le gant s'est connecté en Bluetooth à l'esclave.
  - Chenillard : Indique que le gant est en fonctionnement et recherche de connexion.

```

void low_blink(){
    unsigned long currentMillis = millis();
    if (currentMillis - timer_led >= vitesse_blink) {
        timer_led = currentMillis;
        if(i>10) aller = false;
        if(i<2) aller = true;
        if(aller) i++;
        else i--;
        for(int j=0; j<=NUMPIXELS; j++){
            pixels.setPixelColor(j, pixels.Color(couleur[0]*i,couleur[1]*i,couleur[2]*i));
        }
        pixels.show();
    }
}

void chenillard(){
    unsigned long currentMillis = millis();
    if (currentMillis - timer_led >= vitesse_chenille) {
        timer_led = currentMillis;
        chenille (i,7);
        chenille (i-15,7);
        pixels.show(); // This sends the updated pixel color to the hardware.
        if(i>NUMPIXELS-1) i=0;
        else i++;
    }
}

```

Le code de l'esclave est plus simple puisqu'il s'agit de récupérer les données envoyées par le maître par Bluetooth puis de faire clignoter la LED de 100W en accord avec la fréquence calculée par le maître. Ce code se décompose donc en 2 parties :

- Récupération des données via Bluetooth. Cette partie similaire à la partie Bluetooth du programme maître dans sa construction, la seule différence est que cette fois ci on récupère les données.
- Flash de la LED : C'est ici que l'effet stroboscopique rentre en action. Le programme allume la LED principale pendant un bref instant (500µs) puis éteint cette LED et attend pendant la durée indiquée par le maître. La LED fait donc des flashes lumineux à une fréquence variable en fonction du temps d'attente indiqué par le programme maître

```
//*****bluetooth*****//
char receive;
if (BTSerial.available()){
    receive=BTSerial.read();
    //Serial.write(receive);
    if(receive=='$')
        BTSerial.print("$");    //test de connection
    else{
        if(receive != '\r' && receive !='\n'){ //si on n'est pas en fin de ligne
            if(receiving_data)                //si on doit enregistrer les donnee
                message+=receive;            //ajouter le char au string
            if(receive=='&')                  //si debut de ligne
                receiving_data=true;          //mode enregistrement active
        }
        else {
            receiving_data=false;             //on n'enregistre plus rien
            Serial.println(message);          //debug
            if(message.toInt() !=0)           //on verifie si tout le message est convertible
                dureeSombre=message.toInt(); //conversion en int
            message="";
        }
    }
}

//*****LED*****//
led_state = HIGH; // turn it on
if(dureeSombre!=1)
    digitalWrite(led_pin, led_state); // Update the actual LED
delayMicroseconds(dureeFlash);
led_state = LOW; // Turn it off
digitalWrite(led_pin, led_state); // Update the actual LED
delayMicroseconds(dureeSombre);
```

## DIFFICULTÉES

Le projet ne s'est bien sûr pas déroulé sans complications. Nous avons fait faces à quelques difficultés à la fois sur le plan matériel et sur le plan logiciel.

La première source de difficultés a été l'exploitation des données de l'accéléromètre. L'accéléromètre utilise un bus I<sup>2</sup>C pour transmettre les données aux objets qui lui sont connectés, il a donc fallu apprendre à utiliser la library *Wire* d'Arduino qui facilite la mise en place des protocoles de communication I<sup>2</sup>C. De plus, l'accéléromètre étant sensible au bruit (tremblement de la main et autres), nous avons décidé d'utiliser un filtre de Kalman. Le principe du filtre est assez simple : il s'agit entre autres d'utiliser la vitesse de rotation actuelle pour prévoir le prochain angle de la main. Cependant nous avons eu de la difficulté à exploiter ce filtre, les calculs étant difficiles à suivre. Au final, l'exploitation de l'accéléromètre nous a pris beaucoup de temps.

```
i2cRead(0x3B); //acquisition des données,
accX = ((data[0] << 8) | data[1]);
accY = ((data[2] << 8) | data[3]);
accZ = ((data[4] << 8) | data[5]);
gyroZ = ((data[12] << 8) | data[13]);
```

Nous avons aussi eu quelques problèmes avec le Bluetooth. En effet au début nous voulions utiliser les modules HC-05 et HC-06 mais malgré tous nos efforts, impossible de faire fonctionner correctement les modules.

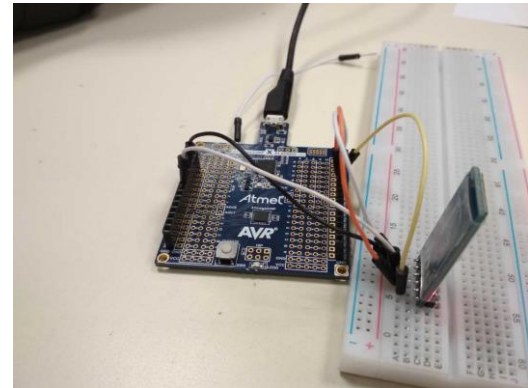


Figure 6 module HC-05

Nous nous sommes donc tournés vers les HM-10, qui se sont avérés être des imitations. Cela ne nous a pas dérangé puisque les modules étaient opérationnels. Il a fallu ensuite produire une connexion stable et efficace, et être capable à tout moment de vérifier l'état de la connexion. Pour cela nous avons établi des règles de connexion :

Les modules s'envoient des '\$' à intervalles réguliers pour vérifier la connexion.

Chaque donnée envoyée est précédée d'un '&' pour être certain qu'il s'agit d'une donnée et non d'une interférence.

Un autre problème avec la Bluetooth est survenu sur le software. En effet pour communiquer avec le module Arduino utilise la librairie *SoftwareSerial*. Cette librairie utilise des interruptions, ce qui génère des conflits avec les fonctions de temps comme `millis()` ou `delay()`. En temps normal ces conflits ne sont pas gênants, mais nous utilisons des temps de l'ordre de la microseconde, et donc ces interruptions deviennent visibles. Cela génère des instabilités dans la fréquence de fonctionnement. Pour régler le problème, nous avons utilisé une librairie alternative sur la partie esclave, qui possède des interruptions plus courtes.

```
#include <SoftwareSerial.h>           → #include <AltSoftSerial.h>
SoftwareSerial BTSerial(11, 10);      AltSoftSerial BTSerial;
```

Parmi les possibilités d'amélioration, nous avons pensé améliorer le design qui est loin d'être parfait. L'aspect visuel des gants pourrait être amélioré en camouflant les composants et les fils, on peut aussi imaginer des fonctionnalités de personnalisation, comme la possibilité de choisir la couleur des LED.

Nous aimerions aussi améliorer le confort des gants. En effet, certains composants tels que les batteries ou le DCDC Converter sont lourds ou encombrants, et certains fils peuvent gêner l'utilisateur. Nous avons fait de notre mieux pour disposer les composants d'une façon ergonomique, mais des améliorations pourraient encore être apportées sur ce point, avec par exemple des batteries moins lourdes.

Par exemple, on pourrait imaginer une « maille » de piles plates. De cette façon, on aurait une alimentation « souple » qui s'adapterait au gant.

Enfin, on aurait aimé trouver ou fabriquer des objets ludiques permettant de montrer l'effet de façon amusante. On pourrait par exemple imaginer une horloge dont les aiguilles tourneraient très vite, des hélices, ou tout type d'objet ayant une fréquence de rotation suffisante. Nous aimerions garder les gants pour pouvoir les améliorer de notre côté.

Globalement, nous pensons que le StroboGlove peut être utilisé pour du divertissement, ou pour une démonstration scientifique de l'effet Stroboscopique.

## CONCLUSION

### Ce que ce projet nous a apporté

Ce projet nous a enseigné beaucoup de choses en ce qui concerne la programmation embarquée : la gestion de la mémoire, de l'espace disponible, de la réactivité du programme, et de la consommation. Mais aussi en ce qui concerne les protocoles de communication I<sup>2</sup>C et la connectivité Bluetooth. Enfin, ce projet a demandé une organisation considérable, et une adaptation aux différents besoins et au budget disponible. Au final nous avons amélioré notre expertise dans la gestion de projet.

### Remerciements

Merci beaucoup aux forums d'Arduino et autres ressources disponibles sur internet qui ont été très utiles pour résoudre nos problèmes.

Merci à Hugo Lavezac pour nous avoir gentiment donné ses gants d'escalade. C'est vraiment sympa.

Merci aussi au SoFab pour nous avoir permis d'utiliser le matériel et d'assembler le projet.

Et surtout un grand merci aux encadrants du projet, Messieurs Pascal Masson et Fabien Ferrero, pour nous avoir fourni le matériel, pour avoir été disponible si besoin, et sans qui ce projet n'aurait pas pu être réalisé.



## SOURCES

---

Le projet qui nous a beaucoup aidé et duquel on s'est beaucoup inspiré :  
<http://www.instructables.com/id/DIY-Time-Control-Machine/>