NEO PROJECT

Problèmatique : Comment peut-on aider une personne aveugle à se mouvoir seule avec de l'électronique à faible coût ?

Projet Neo ? Mais encore ?…

Le projet Neo est un dispositif embarqué sur un bandeau de sport classique permettant de mesurer la distance séparant une personne aveugle des obstacles qui l'entourent et de détecter la présence d'une ou plusieurs personnes face à elle pour faciliter ses déplacements et son interaction avec autrui. Les utilisateurs seront alors en mesure d'être guidés par des sons qu'ils percevront dans une direction précise qu'ils pourront situer. L'utilisateur entend un son venir de la direction de l'obstacle ou du visage qui est face à lui. Nous y reviendrons plus en détail par la suite.

Pourquoi ce projet ?

Le père de Thomas Perrot (partie SIN) s'occupe énormément de personnes handicapées dans son travail et il lui a souvent parlé de la vie des personnes aveugles qu'il a rencontré. Ces dernières rencontrent énormément de soucis liés à leur cécité au quotidien. Nous voulions tous en savoir plus et avons donc décidé de contacter les personnes aveugles concernées. La curiosité a par la suite laissé place à une certaine stupéfaction lorsque nous avons comparé les récits des différentes personnes interrogées. Leurs mésaventures étaient étrangement similaires : Collisions avec des panneaux sur les trottoirs, difficultés à se déplacer seules ou encore des problèmes pour repérer les personnes qui chuchotent ou se taisent naturellement à l'approche de quelqu'un d'étranger à leurs conversations et qui, par conséquent, ne sont plus repérables par les aveugles (faute de sons…).
Tous ces problèmes n'avaient jusqu'à présent que des solutions inaccessibles ou rudimentaires (Matériel électronique coûteux ou canne blanche…), cependant, il paraît très simple d'y remédier très simplement et pour peu cher. Nous ne nous vantons pas de vouloir

rendre la vue mais nous pouvons certainement substituer au mieux ce qu'elle apporte en complétant un autre sens : l'ouïe.

Et l'innovation ?

Neo n'invente rien, il ne réinvente rien. Neo combine et recombine des technologies simples et anciennes dans le but de créer quelque chose de nouveau. Là où des chercheurs sont désireux d'implanter la crème des avancées technologiques dans des projets onéreux, nous utilisons des technologies utilisées dans d'autres domaines comme le cinéma, la musique (sons binauraux), la robotique (détection des obstacles et des visages), le sport (bandeau). Nous voulons exploiter l'intégralité d'un sens de l'utilisateur, pourquoi ne pas en faire de même avec la technologie que nous rencontrons au quotidien ?

Notre démarche :

Nous établissons une liste des problèmes à résoudre. Nous entamons donc des échanges avec les personnes concernées. Nous établissons un cahier des charges en fonction des BESOINS auxquels il nous faut répondre. Nous examinons toutes les techniques existantes nous permettant de répondre à un problème. Nous sélectionnons la plus adaptée à notre projet et nous nous lançons dans le choix des composants.

RAPPEL DES BESOINS :

- Détecter les obstacles environnants afin d'éviter un accident

- Détecter les personnes en face de l'utilisateur pour les repérer quel que soit le niveau sonore

- Communiquer les informations via un dispositif audio

A présent, je pense que nous pouvons commencer.

Détecter les obstacles environnants

C’est sûrement le problème que nous avons le plus facilement résolu !
Celui-ci exige la détection des obstacles afin de les situer dans l’espace. Pour cela, il suffit d’obtenir la distance séparant un point (ici le crâne de l’utilisateur) d’un objet/ d’une personne sur l’intégralité du champs de vision humain, soit environ 180°.
Il nous faudra donc un capteur balayant une large zone ou plusieurs capteurs. La seconde solution semble plus facile à mettre en œuvre et c’est celle-ci que nous choisirons.
Cependant, Comment mesurer une distance avec un matériel électronique sans qu’un contact avec une surface extérieure au système ne soit requise et sans dépasser le budget de 100-150 € dont nous disposons pour toute la partie informatique ? Il faut également que cette solution soit facilement « embarquable ».

Pour cela, intéressons-nous aux technologies actuellement utilisées pour mesurer une distance.

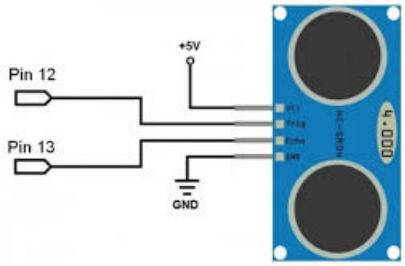
Il existe sur le marché un grand nombre de capteurs de distance : infrarouge (réflectif), laser (par temps de parcours ou par calcul d'angle), physique (règles optiques absolues ou incrémentielles), ou ultrason.

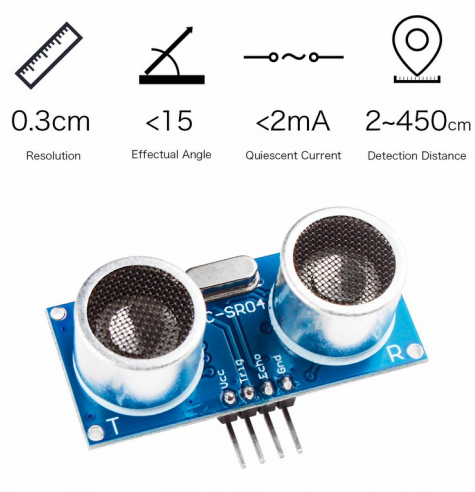
- Les capteurs infrarouges ont l’avantage d’être bon marché et relativement précis. Il faut appliquer une formule complexe pour obtenir une mesure utilisable. De plus, ils sont très sensibles à la lumière ambiante et au coefficient de réflexion lumineuse de la surface en face du capteur. Les vitres nous causeraient également quelques problèmes…

- Un capteur de distance laser (par mesure de temps de parcours) est vendu à un prix avoisinant les 200€, mais fait des mesures à plus de 30 mètres sans problème pour certains modèles. Nous avons pas besoin de mesurer des distances aussi grandes et nous n’allons pas investir un tel budget dans un seul capteur !

-Il nous reste les capteurs à ultrasons.Un capteur de distance à ultrason utilise le même principe qu’un capteur laser, mais en utilisant des ondes sonores (inaudibles) au lieu d’un faisceau de lumière. Ils sont bien moins chers que les capteurs laser ! Contrairement aux capteurs à infrarouge, la lumière ambiante et l’opacité de la surface en face du capteur ne jouent pas sur la mesure.

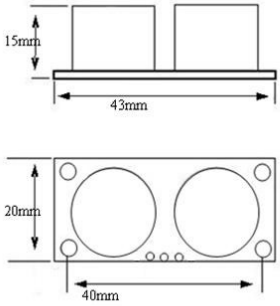
Nous utilisons une carte Arduino pour traiter les données de notre appareil (Hormis les données de détection faciale). Nous allons donc utiliser des capteurs très populaires sur ce support car faciles à mettre en place et peu gourmands en électricité : Les HC-SR04.

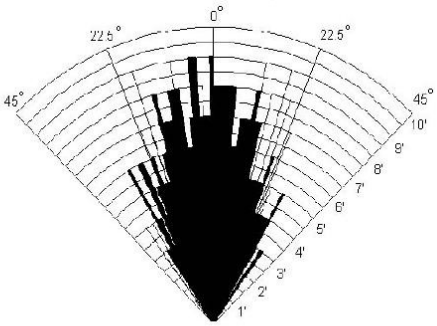




Les schémas ci-dessus montrent les deux broches d’alimentations (Vcc /+5V, Gnd) ainsi que les broches de contrôle de l’émetteur et du récepteur (TRIGGER et ECHO).

On notera également que chaque capteur dispose d’un angle de détection efficace d’environ 30° :





Practical test of performance,
Best in 30 degree angle

Le nombre de capteur utilisés **sur la version finale du projet** sera déterminée plus bas dans ce document.

Détecter les visages

Ici, nous sommes sur un terrain un peu plus hostile. Nous avons eu l'habitude de traiter des données très simples au cours des projets que nous avons dû réaliser au cours de l'année de première et au début de celle de Terminale. Il y a une différence entre le traitement d'informations analogiques fournies par un potentiomètre et le traitement de chaque pixel d'une image afin de reconnaître un visage humain. Nous avons donc décidé d'utiliser des algorithmes nés quasiment en même temps que nous et très utilisés pour ne pas dire qu'ils ont un quasi-monopole sur les systèmes de reconnaissance faciale. Évidemment, la première page d'une simple recherche Google a suffit à nous faire comprendre que cette fois-ci, nous allions devoir nous passer de l'Arduino. Nous avons donc décomposé chaque étape à réaliser pour mettre en œuvre la détection de visages afin d'obtenir des réponses structurées sous deux grands axes de la chaîne d'information ACQUERIR et TRAITER :

I – Acquisition des informations

Ici, pas énormément de solutions possibles...Un capteur de distance, qu'il soit à Infrarouges, à ultrasons où même à laser, ne sera pas suffisant pour fournir des informations précises pouvant être exploitées dans notre détecteur de visages.
Il nous faudra utiliser un dispositif capable de fournir une image assez précise pour que la partie « TRAITER » puisse distinguer un visage humain du reste de l'environnement.

Notre choix s'oriente alors vers l'utilisation une camera.

II – Traitement des informations

Matériel à utiliser ?

C'est là que les choses se compliquent...Nous utilisons une carte Arduino pour le traitement des informations concernant les capteurs de distance mais après quelques recherches sur le sujet, celle-ci ne s'avère pas à adaptée à la détection de visages. De plus, le temps et les

connaissances dont nous disposons ne nous permettront pas de réaliser et un algorithme et un programme de détection de visages ; Nous avons donc recherché un autre composant, similaire à la carte Arduino par son format, mais beaucoup plus rapide et performant sur lequel la détection faciale a déjà été adaptée (avec des bibliothèques de programmation dont l'utilisation reste similaire à celle que nous avons déjà eu l'occasion d'utiliser sur Arduino).

En cherchant les différentes solutions existantes sur Internet, nous avons trouvé une bonne poignée de logiciels pour PC (exploitant les systèmes Windows, Linux etc...) capables de réaliser cette tâche. Cependant, il paraissait compliqué d'embarquer un ordinateur supportant une distribution Windows ou Linux à faible coût...avant 2012 et la sortie des ordinateurs mono-carte Raspberry Pi et de leurs nombreuses déclinaisons capables d'exécuter une multitude de versions de Linux (les Pi possèdent même des adaptions dédiées du système au pingouin).
Il bénéficie également de sa propre « Pi Camera », plus facile à mettre en place qu'une Webcam USB standard.
(La camera se connecte directement au Raspberry Pi via une nappe de connexion et est préconfigurée par défaut sur les dernières versions des distributions Linux officielles du Raspberry).

Et la programmation ?

Le hasard faisant bien les choses, nous avons également trouvé une bibliothèque libre de droits compatible avec le Raspberry Pi et ***Linux Raspbian*** comportant des algorithmes complexes de détection d'objets/visages (fonctionnement) conçue pour la programmation en Python.

```
pi@raspberrypi ~ $ workon cv
(cv)pi@raspberrypi ~ $ python

[GCC 4.9.1] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.0.0'
>>>
```

Nous considérerons cependant le Raspberry comme une solution de dernier recours en raison des quelques contraintes qu’il impose (taille, consommation 2A pour 5V...)

Cependant, pour nos essais en atelier, nous avons expérimenté l'utilisation du Raspberry Pi et d'une Pi camera pour travailler notre maîtrise des outils de détection faciale.
Ce choix ne sera pas définitif.

Nous avons préféré un autre composant...

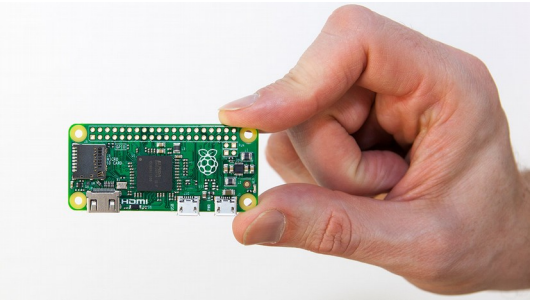
Détection faciale : Solution finale

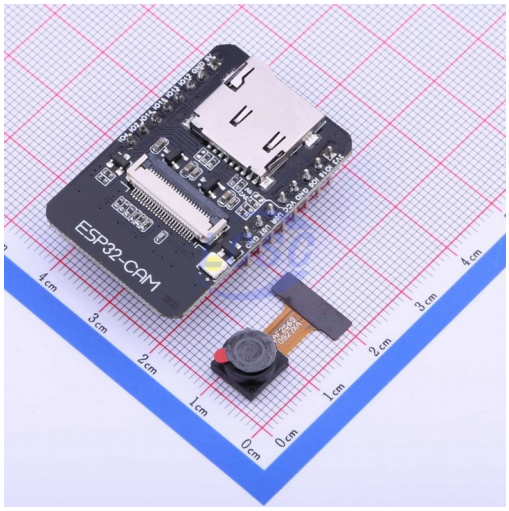
Une semaine après avoir essayé le Raspberry Pi, nous avons trouvé un module basé sur le microcontrôleur « ESP 32 » de chez Espressif. Ce concurrent d'Arduino a rendu son composant utilisable avec la distribution le logiciel « Arduino IDE » et l'a surtout rendu plus rapide…
Là ou l'Arduino travaille sur 10 bits avec une cadence maximum de 16 MHz, l'ESP travaille sur 32 bits et possède une vitesse de calcul (en crête) de 240 MHz. Cette nouvelle configuration permet de traiter les images. Espressif a donc développé un logiciel de programmation dédié à la détection/reconnaissance de visages sur son microcontrôleur. Nous allons donc nous contenter d'envoyer les coordonnées d'un visage sur une image à l'Arduino (via une liaison série).

ESP 32 monté sur une carte « ESP EYE » dotée du connecteur nécessaire à une caméra compatible. Nous en recevrons un exemplaire d'ici quelques jours.

La petite taille de ce composant, sa faible consommation en électricité (20 mA pour 5V sans LED de flash) et sa facilité d'utilisation le rendent plus adapté à ce projet qu'un Raspberry Pi.

L’ESP n’a rien à envier de la taille d’un RPI Zero





Communiquer les informations via un dispositif audio

Bien qu’ayant déjà en tête l’idée de la façon dont nous allions diffuser les sons, nous avons comparé les différentes techniques utilisées :

I – Méthode de communication sonore

Ici, notre choix favorisera une communication intuitive pour l'utilisateur. La communication avec ce dernier devra être réalisée de façon discrète et communiquer des informations précises sur la position des obstacles. L'utilisation d'un haut parleur classique diffusant le son dans un large périmètre ou d'un « buzzer » piézo-électrique ne semble donc pas adaptée à notre problème.
Nous préconiserons donc l'utilisation d'un casque audio stéréo ou non en fonction de la méthode de communication qu'il nous reste à choisir.

Plusieurs solutions s'offrent alors à nous :

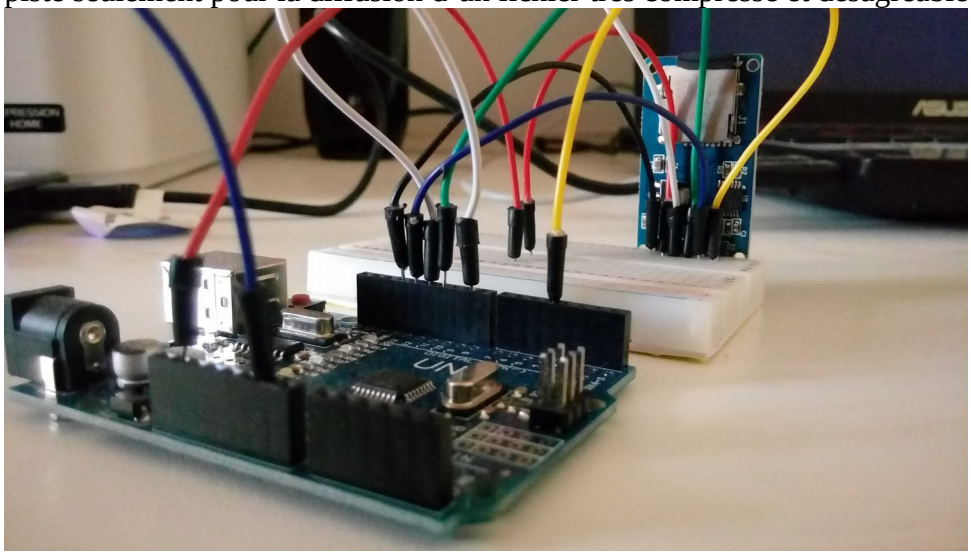
- Une communication vocale phrasée (Ex : « Obstacle à 2 m droit devant... »)
- Un langage prédéfini que l'utilisateur doit préalablement maîtriser (Ex : Suite de « bip »)
- Un son modulé de façon à être perçu dans une direction précise (Audio 3D/Son binaural)

La première solution ne semble pas convenir à l'appareil. En effet, il est primordial de prévenir **rapidement** l'utilisateur (aveugle !) de la présence d'un obstacle pour éviter les accidents.

Nous favorisons la méthode la plus intuitive déjà évoquée en début de document, l’audio 3D. Cette méthode procure un « effet de spatialisation sonore » en procurant l’impression au porteur d’un casque (indispensable) que les sons sont diffusés à gauche, à droite, devant ou derrière lui.

Nous avons essayé de lire des fichiers audio directement sur Arduino via une carte SD mais les faibles capacités du matériel empêche la lecture de fichiers stéréo. Il ne permet de diffuser qu’une

piste seulement pour la diffusion d’un fichier très compressé et désagréable à l’oreille.



SD_PLAYER §

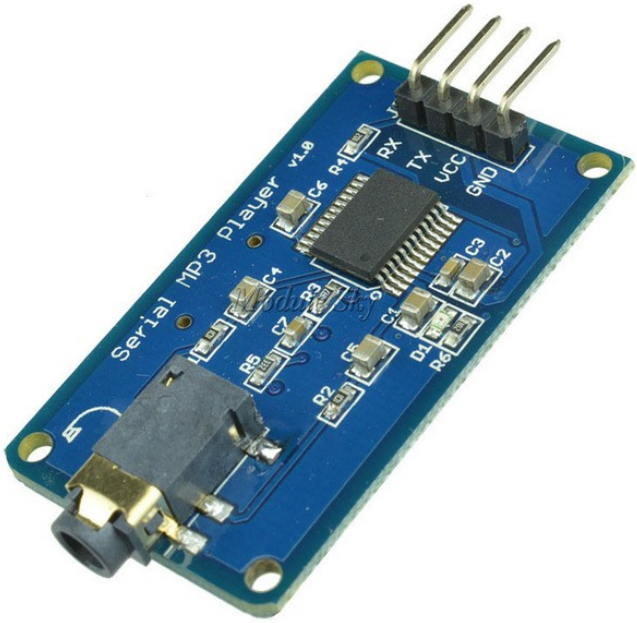
```
//PROGRAMME DE TEST DE LECTURE DE FICHIERS AUDIOS POUR "SODIAL SD module".

#include <SD.h>
#include <TMRpcm.h>

#define SD_ChipSelectPin 4                //broche CS

TMRpcm tmrpcm;                            //nouvel objet (bibliothèque TMRpcm)



void setup(){

  tmrpcm.speakerPin = 9;                  //haut-parleur = broche 9


  if (!SD.begin(SD_ChipSelectPin)) {

    return;
  }

  tmrpcm.setVolume(6);                    //volume à 6    (plage 0-7)

  tmrpcm.quality(0);                      //Normal = 0 ,   Sur-échantillonage = 1

  tmrpcm.play("test.wav");                //Jouer le son "test.wav" de la carte SD
}

void loop(){}
//Vide pour ce test
```

Programme de lecture sur un simple module SD

Audio 3D : Solution finale (un peu plus de technique!)

Le traitement des fichiers audios ne pouvant pas être assuré par une Arduino seule, un autre petit microcontrôleur faible consommation (20 mA pour 5V) sera utilisé, il s’agit du Catalex YX5300 !

Le voici >>>>>>>>>>>>



Le module de Catalex est doté du circuit intégré YX5300 permettant un traitement de fichiers audio lourds avec un échantillonnage max de 48 kHz sur 24 bits contre les 16 kHz sur 8 bits seulement de l'ATMEGA328 de nos bonnes vieilles Uno...

En ce qui concerne le son, la carte est équipée d'un port mini-jack femelle prêt à l'emploi. Nous n'aurons pas l'utilité d'un système d'amplification sonore externe puisque le son sera diffusé sur un casque. En ce qui concerne le contrôle de la carte depuis notre Arduino, il faudra passer par une simple communication série (broches RX et TX).
C'est ici qu'on s'amuse. La documentation énumère les commandes disponibles sur YX5300 en hexadécimal pour nous faciliter la vie :

	7E FF 06 **19** 00 00 **01** EF	Close single cycle play
[Set DAC]	7E FF 06 **1A** 00 00 **00** EF	Start up DAC output
	7E FF 06 **1A** 00 00 **01** EF	DAC no output
[Play with volume]	7E FF 06 **22** 00 **1E 01** EF	Set the volume to 30 (0x1E is 30) and play the first song
	7E FF 06 **22** 00 0**F 02** EF	Set the volume to 15(0x0f is 15) and play the second song

Command	Command bytes without checksum(HEX)	Remark
[Next Song]	7E FF 06 **01** 00 00 00 EF	
[Previous Song]	7E FF 06 **02** 00 00 00 EF	
[Play with index]	7E FF 06 **03** 00 **00 01** EF	Play the first song
	7E FF 06 **03** 00 **00 02** EF	Play the second song
[Volume up]	7E FF 06 **04** 00 00 00 EF	Volume increased one
[Volume down]	7E FF 06 **05** 00 00 00 EF	Volume decrease one
[Set volume]	7E FF 06 **06** 00 **00 1E** EF	Set the volume to 30 (0x1E is 30)
[Single cycle play]	7E FF 06 **08** 00 **00 01** EF	Single cycle play the first song
[Select device]	7E FF 06 **09** 00 **00 02** EF	Select storage device to TF card
[Sleep mode]	7E FF 06 **0A** 00 00 00 EF	Chip enters sleep mode
[Wake up]	7E FF 06 **0B** 00 00 00 EF	Chip wakes up
[Reset]	7E FF 06 **0C** 00 00 00 EF	Chip reset
[Play]	7E FF 06 **0D** 00 00 00 EF	Resume playback
[Pause]	7E FF 06 **0E** 00 00 00 EF	Playback is paused
[Play with folder and file name]	7E FF 06 **0F** 00 **01 01** EF	Play the song with the directory: /01/001xxx.mp3
	7E FF 06 **0F** 00 **01 02** EF	Play the song with the directory: /01/002xxx.mp3
[Stop play]	7E FF 06 **16** 00 00 00 EF	
[Cycle play with folder name]	7E FF 06 **17** 00 00 **01** EF	01 folder cycle play
[Shuffle Play]	7E FF 06 **18** 00 00 00 EF	
[Set single cycle play]	7E FF 06 **19** 00 00 **00** EF	Start up single cycle play

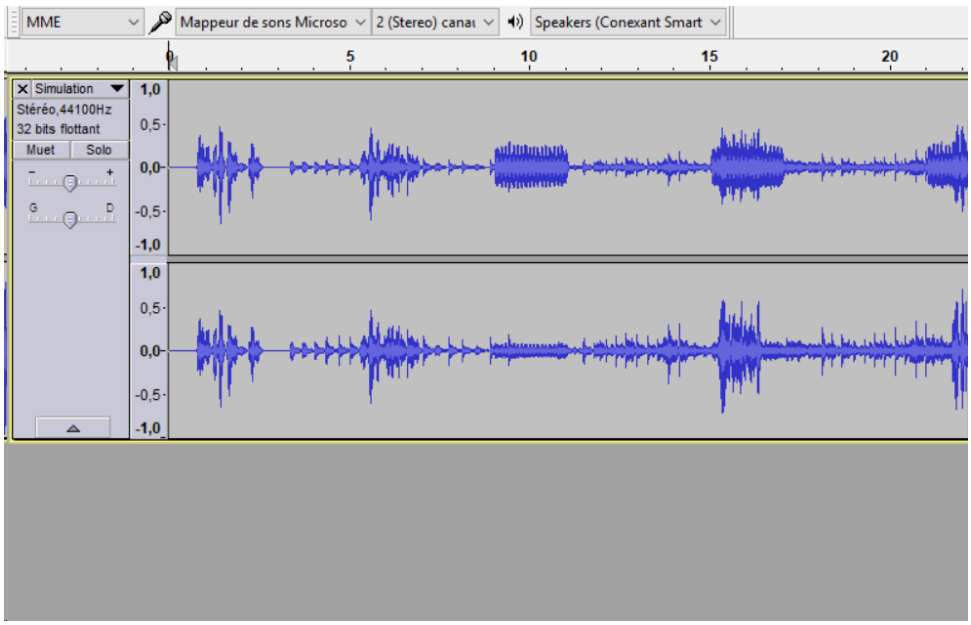
Nous retrouvons également la structure détaillée de chaque commande envoyée (bit de start, nombre de bits de la donnée etc…) dans la documentation.
Elle est nécessaire pour comprendre le fonctionnement précis de l'écriture d'une donnée (le tableau est cependant très bien réalisé et permet de comprendre rapidement la méthode de communication avec la carte).
Bien évidemment, nous n'utiliserons pas toutes les instructions disponibles.

A présent, penchons-nous un peu sur le son lui-même. Le système se doit d'utiliser une communication audio intuitive. Comme vous le savez déjà, je me suis tourné vers une technologie très intéressante : L'audio 3D.

RAPPEL : elle permet à l'utilisateur d'un casque audio (dans notre cas) d'entendre les sons dans une direction précise sur un champs de 360° tout autour de lui (la perception des sons venant du haut et du bas n'étant pas très efficace avec les casques stéréo standards).

Maintenant que tout est clair, revenons à notre système. Nous pouvons diviser les informations communiquées par le son diffusé en deux données : La *direction* et la *distance.*
La *direction* définie par un angle compris entre 0 et 360° (cas général) sera traitée indépendamment de la *distance*.
Notre champs de détection des obstacles est étendue sur 180°. Nous décomposerons cet angle total en un nombre n de zones.
Chaque zone disposera d'un fichier identique (ou non d'ailleur…) à celui des autres avec pour seules différences un infime décalage entre les pistes "Droite" et "Gauche", une modification de la réverbération et une égalisation du volume sonore, le tout constituant un ensemble de données précalculées (voir suite) qui une fois restituées, modifieront la perception du son.

Pour cela, j'utiliserai le logiciel *Audacity* ainsi qu'un plugin me permettant d'effectuer toutes les modifications sonores sur ordinateur.



Démonstration d'un son binaural trouvé sur YouTube (casque audio stéréo requis) :
https://youtu.be/8IXm6SuUigI
NOUS N'EN SOMMES PAS LES AUTEURS

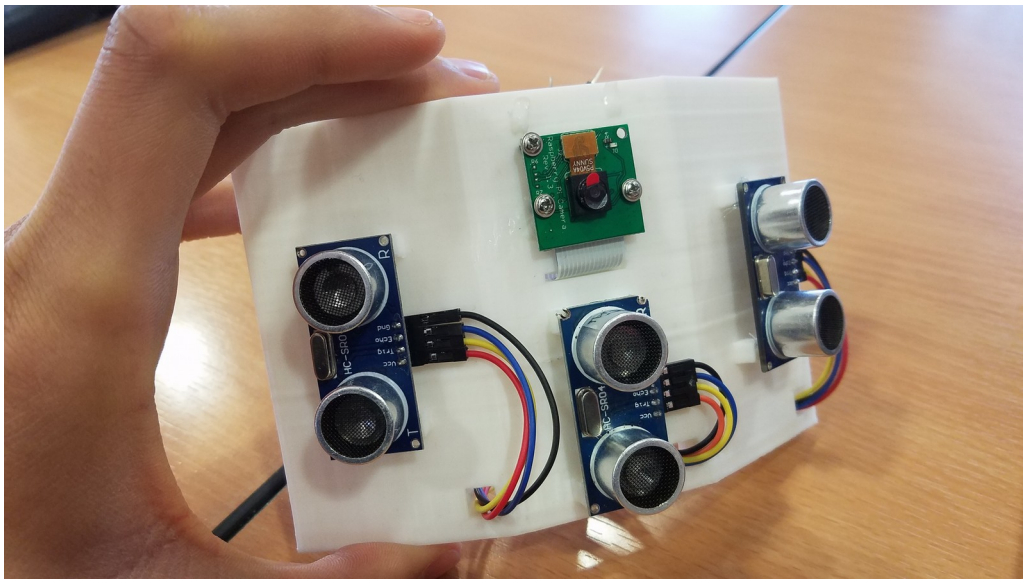
Voilà pour le premier paramètre.

En ce qui concerne le second, nous ferons simplement varier le volume sonore pour simuler une distance plus ou moins importante entre l'obstacle et l'utilisateur. Cette opération sera réalisée par le YX5300 grâce à une commande qui nous permettra de choisir entre 30 paliers de réglage du volume.
Le programme choisira judicieusement le palier à utiliser en fonction de la distance mesurée par les capteurs à ultrasons HCSR04.

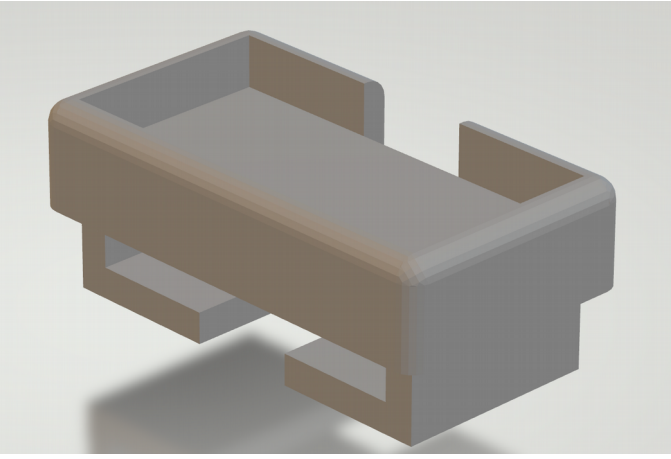
[Set volume]	7E FF 06 **06** 00 **00 1E** EF	Set the volume to 30 (0x1E is 30)

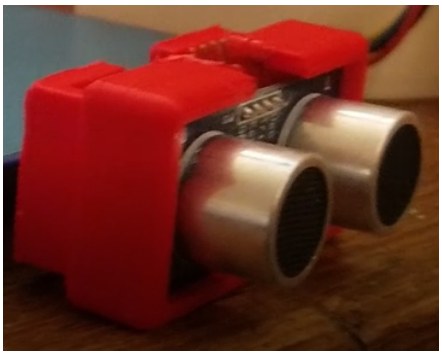
La portée des HCSR04 étant d'environ 4 mètres, nous définirons qu'une variation d'une unité de volume sonore correspondra à une distance différente de 13 centimètres par rapport à celle initialement mesurée.

Prototype n°1 : L'accordéon



Ce prototype est un support que nous avons réalisé pour tester toutes les solutions précédemment citées. Il regroupe la détection d'obstacles, de visages (Raspberry Pi) ainsi que la communication sonore. Nous remplacerons bientôt le Raspberry par l'ESP 32. Le montage sur bandeau sera bientôt réalisé avec les pièces nouvellement imprimées en 3D :





Une fois imprimé !

Nous avons actuellement des résultats très concluants. Nous savons désormais exactement comment mener à bien ce projet ambitieux et attendons avec impatience l'arrivée des composants que nous posséderons d'ici quelques jours. Il ne nous reste plus qu'à monter la nouvelle caméra et assembler les éléments de chaque discipline entre eux. Nous serions prêts à vous présenter l'objet fini d'ici Avril.
Merci d'avoir lu :)

Toute l'équipe du projet Neo.

Mis en forme et rédigé par Thomas Perrot et Benjamin Fourcroy.
Participations actives au sein du projet : Benjamin Fourcroy, Lucas Rodet, Ollivier Clémence, Françoise Jamart, Thomas Perrot.

Lycée Gustave Eiffel, Dijon.