Documentation Projet Robot : Piano-Cocktail

1 DESCRIPTION

Le projet Piano-Cocktail est un piano permettant de servir des cocktails selon une séquence de notes jouée. Si celle-ci est fausse, le joueur ne recevra rien. Dans le même ordre d'idée, si le joueur n'est pas majeur, celui-ci ne recevra pas de cocktail non plus.

2 ARBORESCENCE DES FICHIERS

La Raspberry Pi contient :

- AgeRecog
  - gad.py
  - opencv_face_detector.pbtxt
  - opencv_face_detector_uint8.pb
  - age_deploy.prototxt
  - age_net.caffemodel
  - gender_deploy.prototxt
  - gender_net.caffemodel
- cocktail_melo
  - cocktail_melo.py
  - Code.ino
  - Light.py
- MIDIControl
  - notes
  - midi_test.py
- textToSpeech
  - Speak.py
  - image_analyse.jpg
  - main.py

3 EXPLICATIONS DES FICHIERS

AgeRecog

- gad.py : on retrouve dans ce fichier toutes les fonctions liées à la reconnaissance faciale et de l'âge.
    - isAgeOk : détecte si la personne est adulte ou non
      Argument :
        - range : l'intervalle d'âge à tester
    - takePic : prend la photo et la sauve entant que *img_analyse.jpg*
    - highlightFace : permet d'isoler les visages des personnes de la photo
      Arguments :
        - frame : la photo que l'on souhaite analyser
        - net : le réseau neuronal des visages échantillonnés
        - conf_threshold : de base à 0.7, représente le seuil de confiance pour déterminer si le visage entre dans un certain intervalle d'âge

      Renvoie :
        - faceBoxes : une liste de liste contenant les 4 coins de chaque cadre encadrant les visages
        - frameOpencvDnn : renvoie la photo reçue avec les visages encadrés
    - detectAge : détecte l'âge du visage

- opencv_face_detector.pbtxt : fichier protobuf (en format texte), contient un graphe de définition et les entrainements du modèle.
- opencv_face_detector_uint8.pb : fichier protobuf (en format binaire).
- age_deploy.prototxt : décrit la configuration du réseau neuronal de pour les âges.
- age_net.caffemodel : définition des états internes des paramètres des couches du réseau pour les âges.
- gender_deploy.prototxt : décrit la configuration du réseau neuronal de pour les genres.
- gender_net.caffemodel : : définition des états internes des paramètres des couches du réseau pour les genres.

cocktail_melo

- cocktail_melo.py : ce fichier gère la comparaison des séquences et aussi l'envoie entre arduino et raspberry pi
    - sendCocktail : permet d'envoyer l'id du cocktail à l'Arduino.
    - compare_melo : compare la séquence obtenue du clavier avec celles encodées et renvoie l'id spécifique d'un cocktail
    - waitResponse : permet d'attendre un bit pour par exemple le fait que le bouton s'enfonce.
    - sendMelo : envoie la mélodie du cocktail choisi par l'utilisateur à l'Arduino qui lui actionne le jeu de lumière.

MIDIControl

- midi_test.py :
    - note_lettre : permet de transformer les chiffres envoyés par le clavier MIDI en notes (A, A#, B, C, C#, D, D#, E, F, F#, G, G#). De 0 à 11, chaque nombre est lié à une note et leurs multiples seront liées à la même note mais une octave au-dessus à chaque fois
    - play_piano : permet de jouer les notes (elles sont jouées sur le haut-parleur)
      Renvoie :
        - note_list : on renvoie simplement la liste des notes qui ont été jouées et ce, après 7 secondes.
- notes : contient toutes les notes pour plusieurs octaves (de 2 à 5)

textToSpeech

- speak.py :
    - speak : comme cette fonction l'indique, elle a pour but de dire, via le haut-parleur, ce qu'elle obtient en argument. Elle utilise la librairie *festival.*
      Argument :
      sentence : la phrase à dire.

Code arduino:

- *float* **computeLap**(*int* ml) : méthode qui prend en paramètre un nombre de ml et retourne le nombre de tour qu'un moteur devra faire pour avoir ce ml
- *void* **serveDrink**(*int* tab[]) : méthode qui prend en paramètre une liste contenant la composition d'un cocktail et actionne les moteurs pour faire ce cocktail.
- void **Motor**(float lap, int motor) : méthode qui prend en paramètre un nombre de tour à faire et le numéro d'un moteur pour ensuite actionner ce moteur en question pour qu'il fasse le nombre de tour désiré.
- *void* **Serve**(*int* cocktail) : méthode qui prend en paramètre le numéro d'un cocktail et sert ce cocktail.
- *void* **play_melo**() : méthode qui allume les leds du néopixel en fonction des numéros de la liste global myMelo.
- *void* **turnoff_neopixel**() : méthode qui éteint les leds du néopixel.

4 REPRODUCTION

Si tout comme nous, vous vous êtes déjà demandé : « Mais comment puis-je apprendre le piano tout en me servant un cocktail ? » alors vous êtes au bon endroit ! La réponse, évidente, à cette question toutefois existentielle nous est apparue. Nous avons donc décidé de développer un moyen ludique, mais efficace, d'apprendre le piano. Et quoi de mieux comme motivation qu'un bon cocktail servi pour vous ? La combinaison de ces deux éléments nous donne le Piano-Cocktail ! Ou comment apprendre à jouer vos chansons favorites en se servant un cocktail.

Dans les parties qui suivent, vous trouverez comment reproduire le projet et ses technologies. Pour commencer une rapide revue du matériel nécessaire. La section suivante passe en revue les différents modèles 3D utilisés. Ensuite, les pompes péristaltiques, technologie utilisée pour verser les liquides est passée en revue. Dans un premier c'est leur assemblage qui est expliqué. Ensuite une courte explication sur les drivers et moteurs précède l'explication sur le calibrage et la programmation de ces pompes.

Ensuite, tout ce qui est relatif à la Raspberry Py est expliqué. Vous y trouverez toutes les dépendances à installer, les programmes utilisés. Comment les utiliser et comment modifier certaines partie du code. Vous verrez également comment utiliser la reconnaissance d'âge et toute les commandes liées au piano. Vous découvrirez également comment connecter la Rasp et l'Arduino. L'installation et l'utilisation des LED Neopixel est aussi décrite. Finalement vous trouverez les plans de montages pour vous montrer comment reproduire le projet.

Vous pouvez trouver les différents fichiers sur le Git du projet.

4.1 MATÉRIEL

Pour reproduire ce projet, vous aurez besoin de :

- Une carte Raspberry Pi (nous avons utilisé le modèle 3B+)
- Une carte Arduino
- 5 moteurs pas-à-pas
- Deux baffles
- Un amplificateur pour baffle
- Une caméra pour Raspberry pi
- Un piano avec sortie MIDI USB
- Un bouton poussoir
- 30 roulements (6 par pompe)
- Des leds type neopixel

Electronique

- 5 drivers moteurs
- Des composants que vous pourrez retrouver dans le document BOM

Structure

- Une planche de bois de 2.44m sur 1.22m
- 1 à 2 tasseaux
- Des vis

- 2 charnières
- 2 poignées

Au niveau des outils, il est nécessaire d'avoir :

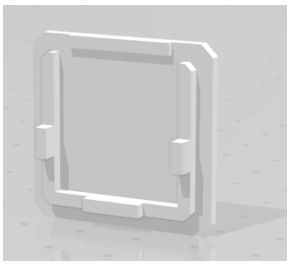
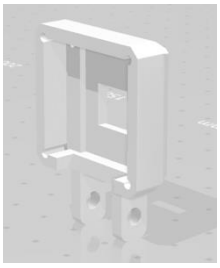
- Une imprimante 3D (il existe des services si vous n'en avez pas)
- Une visseuse
- De quoi couper les planches (scie-sauteuse)
- De la colle à bois
- Fer à souder + fil de soudure
- De la peinture (pour le style)

4.2 IMPRESSION 3D

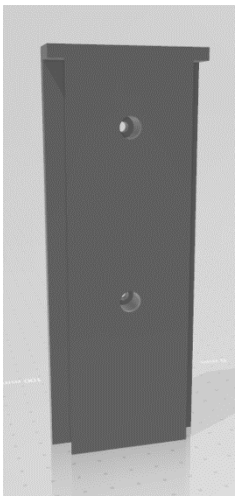
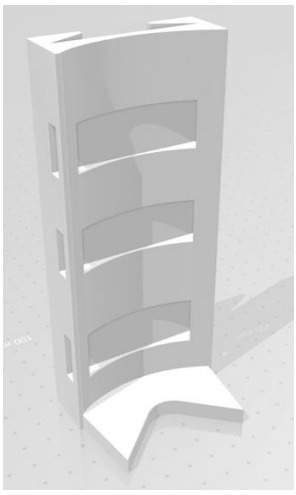
Pour les impressions 3D, tous les fichiers utilisés ou modélisés sont retrouvables au même endroit que ceux de l'Arduino et de la Raspberry.

Vous y retrouverez :

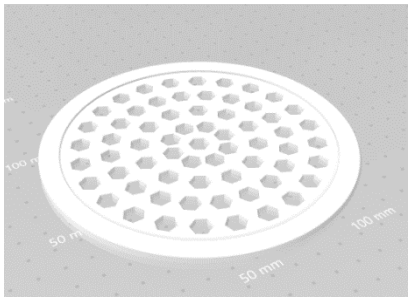
- Le support pour la caméra en 3 parties

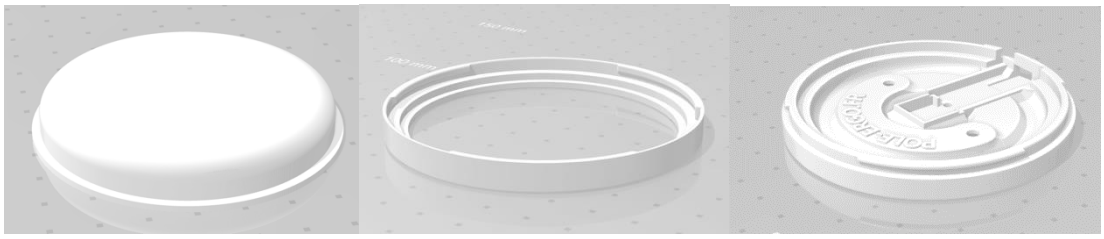
- Les supports de bouteilles en 2 parties

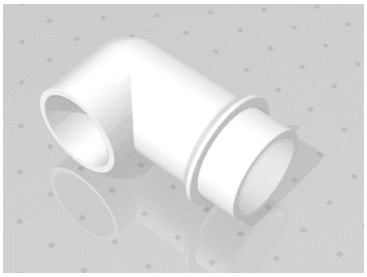
- Les grilles des baffles à redimensionner en fonction de vos modèles



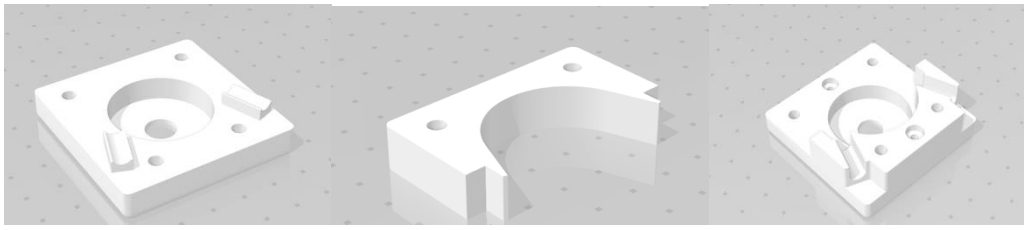
- Le bouton en 3 parties



- Le « robinet »



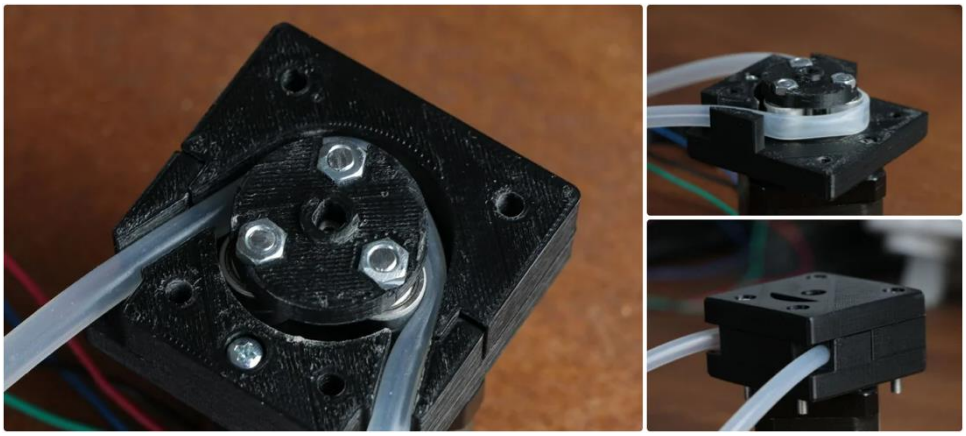
- Les pompes péristaltiques en 5 parties



Les modèles ont été imprimés en couche de 0.2 à 0.3 avec un remplissage de 20%. Il faut activer les supports durant l'impression seulement pour le robinet et les pompes.

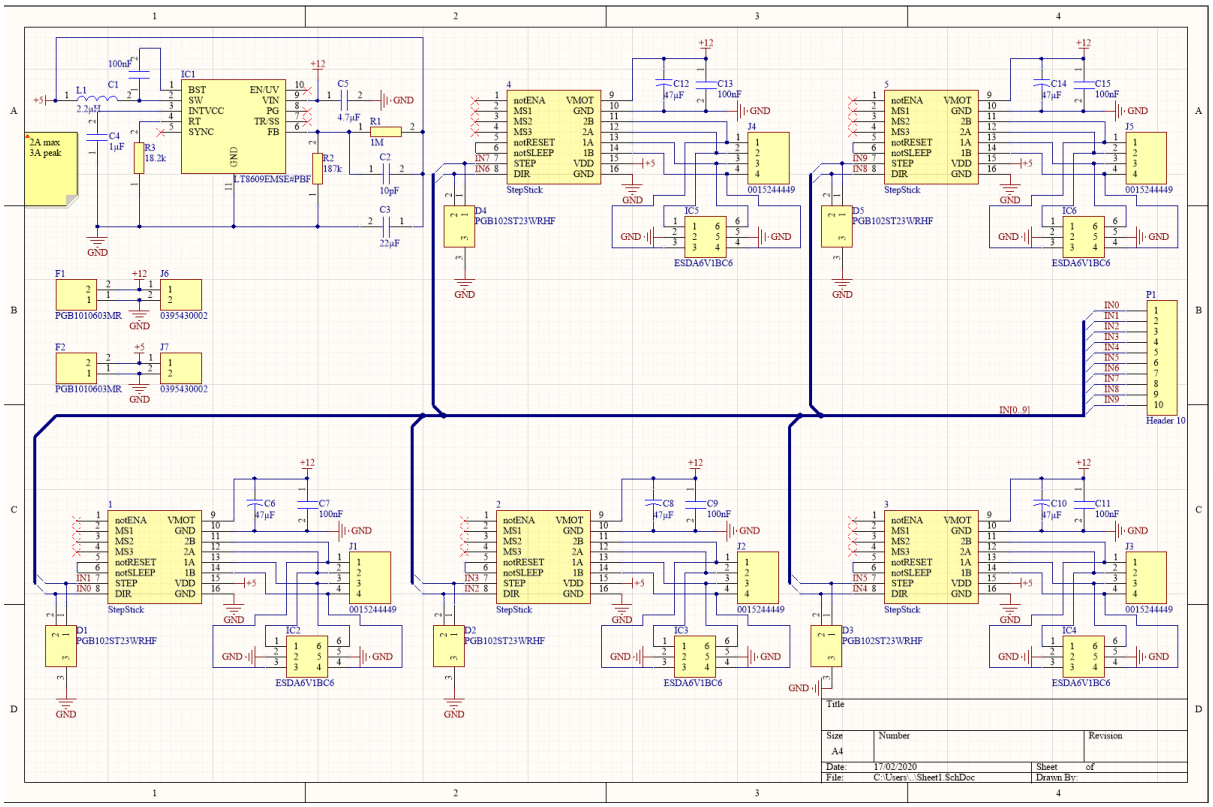
4.3 ASSEMBLAGE DES POMPES

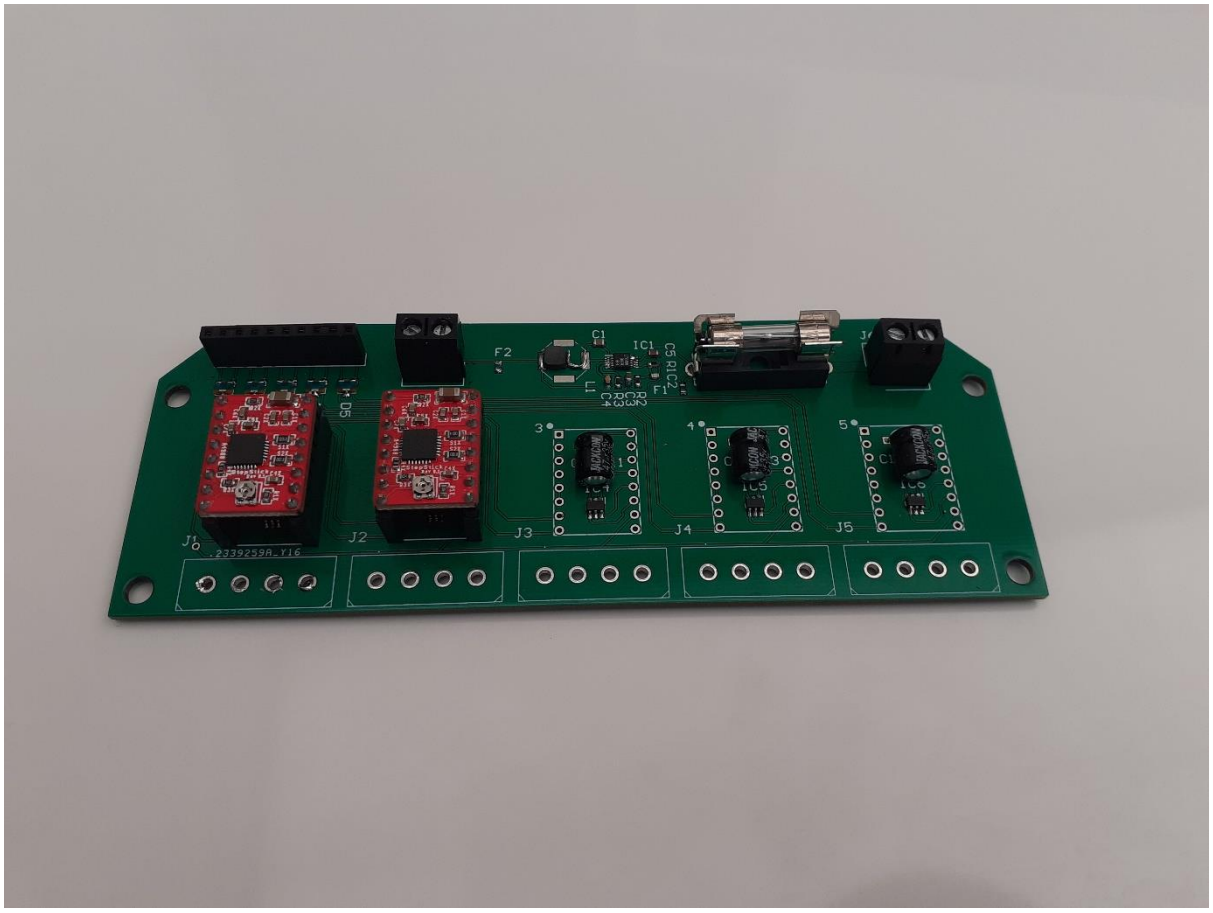
Le modèle retenu après plusieurs tests est le modèle de Great Scott composé de 5 composants imprimés en 3D. Il est nécessaire d'avoir des vis et écrous de type M4 soit 7 écrous, 3* M4x20,4* M4x5, 4* 4Mx25 en vis et enfin 6x roulements à billes 13x4x5. On retrouve l'axe du moteur passant au travers de notre disque composé des roulements. L'assemblage est assez simple, il suffit d'avoir son moteur et d'ensuite empiler et visser les différentes pièces comme sur les photos ci-dessous.



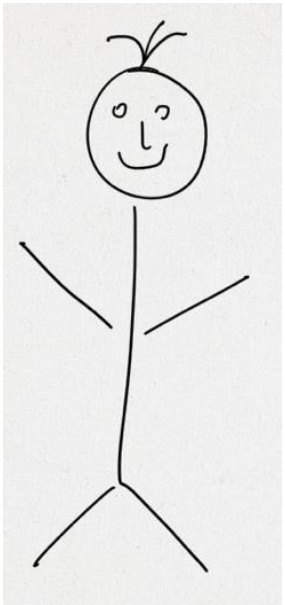
Instructable de Great Scott : https://www.instructables.com/id/DIY-Peristaltic-Pump/

4.4 ASSEMBLAGE DU DRIVER MOTEUR





4.5 UTIILSATION ET TEST DES POMPES



Lui c’est Bob.

Bob n’a pas testé ses pompes à la tension du projet.

Bob a du tout recommencer.

Ne soyez pas comme Bob, testez vos pompes à la bonne tension.

Une fois les pompes assemblées et connectées aux drivers, un petit coup de pouce de l'Arduino est nécessaire. Il est important de déterminer le débit de vos pompes. Pour ce faire, codez d'abord une fonction pour faire tourner les pompes. Cette dernière doit prendre en paramètre un nombre de tours. Étant donné que les moteurs utilisés ont 200 pas par tour, leur précision sera de 0,005 tour. Vous pouvez donc vous permettre d'entrer un nombre flottant de tour. Pour effectuer vos tests, je vous recommande d'utiliser le script « StepperLap » que vous trouverez en cliquant ici. Ce script vous permet d'entrer le nombre de tour que vous désirez effectuer via la commande série. La fonction *rotateLap()* converti le nombre de tours en nombre de pas, change le sens de rotation si le nombre est négatif et fait tourner le moteur du nombre de pas. Vous trouverez également une fonction pour faire tourner les moteurs selon leurs angles. Cette dernière est précise à 2 degrés prêts. Pour l'utiliser il vous suffit de changer la ligne « rotateLap(lap); » dans la fonction *loop()* par « rotateAngle(lap); ». Il ne vous reste plus qu'à trouver la fonction qui correspond à vos pompes en la testant. Pour vous faciliter la tâche, vous pouvez consulter nos tests dans ce fichier Excel pour ensuite remplir votre propre tableau vierge. Le calcul de la fonction est déjà préenregistré.

Maintenant que vous avez votre fonction il est temps de l'implémenter dans le code final. Votre fonction devrait ressembler à :

$$y = 0{,}5141x + 0{,}3393$$

x étant le nombre de tour et y la quantité en ml. Puisque dans notre cas nous connaissons la quantité et cherchons le nombre de tour il faut légèrement modifier la fonction :

$$x = \frac{y - 0{,}3393}{0{,}5141}$$

Afin de ne pas nous tromper dans les calculs, nous avons décidé de l'entrer comme tel dans le code et de laisser l'Arduino calculer de lui-même. Notez qu'une précision à 0,01 pourrait entièrement suffire. Maintenant que nous avons déterminer et retranscrit correctement la loi qui régis nos pompes, il est temps d'en faire une fonction. Dans le fichier Piano_Opti.ino, la loi est retranscrite dans la fonction *computeLap()*.

Désormais nous avons une fonction qui nous permet de calculer le nombre de tour à partir de la quantité et nous savons faire tourner un moteur. Il est temps de créer un fonction pour servir un cocktail. Nous avons décidé de stocker les cocktails sous forme d'Array. Leur taille correspond au nombre de pompe. L'élément 0, contient la quantité que la première pompe doit délivrer, l'élément 1 pour la pompe 2 et ainsi de suite. Afin de ne pas faire de duplication de code, nous allons créer une dernière fonction avant d'effectivement écrire celle qui servira les cocktail. La fonction *motor()*. Elle prend un nombre de tours et un numéro de pompe en paramètre. Le switch nous permet de choisir le bon moteur avec le numéro de la pompe. Le reste de la fonction est une copie de celle utilisée pour les tests. Nous pouvons donc maintenant créer la fonction *serveDrink()*. Elle prend l'Array d'un cocktail en paramètre. Elle parcourt cette Array et pour chaque élément appelle la fonction *motor()*.

N'oubliez en début de code de déclarer les pins *step* et *dir* des différents moteurs ainsi que de les référencer comme output dans le *setup()*.

4.6 INSTALLATION DES DÉPENDANCES

Pour que l'ensemble des codes fournis fonctionne, il vous faudra travailler avec Python 3. Les librairies à installer sont donc :

(Petit rappel : pour les librairies qui s'installe via python, faites bien attention à installer avec **`pip3`**. Si vous ne les trouvez pas avec pip, utilisez **`sudo apt-get`**)

- pygame
- opencv (cv2)
- numpy
- serial
- festival
- (D'autres librairies seront encore à installer[1])

Vous aurez également besoin des librairies données ci-dessous mais en principe, elles sont préinstallées sur l'os Raspbian Buster :

- time
- RPi.GPIO
- argparse
- os
- math

Concernant l'arduino il faudra installer :

- La bibliothèque "Adafruit_NeoPixel" pour le disponible la gestion de bibliothèque

4.7 OPENCV ET RECONNAISSANCE D'ÂGE

Si l'installation de opencv a été faite correctement, vous allez pouvoir tester le code du fichier de reconnaissance d'âge.

Pour ce faire :

1. Connectez la caméra au bon endroit
2. Placez-vous dans le dossier RobotProject (**`cd RobotProject`**)
3. Lancez python3
4. Importez le fichier gad grâce à **`from AgeRecog import gad`**
5. Testez la reconnaissance d'âge via **`print(gad.detectAge())`**

Voici le résultat que vous devriez avoir :

```
>>> from AgeRecog import gad
>>> print(gad.detectAge())
Gender: Male
Age: 15-20 years
15-20
```

[1] Au point concernant l'installation MIDI

4.8 INSTALLATION MIDI ET TEST DU PIANO

Pour que le piano fonctionne avec notre raspberry, l'installation est principalement issue du lien ci-dessous :

https://ixdlab.itu.dk/wp-content/uploads/sites/17/2017/10/Setting-Up-Raspberry-Pi-for-MIDI.pdf

Voici les éléments principaux à configurer :

Setup de la raspberry comme un Gadget USB OTG

- Mettre le driver USB à dwc2

```
echo "dtoverlay=dwc2" | sudo tee -a /boot/config.txt
```

- Activer le driver dwc2

```
echo "dwc2" | sudo tee -a /etc/modules
```

- Activer le driver ***libcomposite***

```
echo "libcomposite" | sudo tee -a /etc/modules
```

- Activer le gadget MIDI

```
echo "g_midi" | sudo tee -a /etc/modules
```

Cette première partie permet de transformer la raspberry comme étant un gadget USB On-The-Go, ce qui permet l'échange de données entre deux appareils sans que l'un soit hôte.

Ensuite, il faut créer le script de configuration.

- Création du fichier

```
sudo touch /usr/bin/midi_over_usb
```

- Le rendre executable

```
sudo chmod +x /usr/bin/midi_over_usb
```

Editer ce nouveau fichier et y coller les lignes qui suivent :

```
cd /sys/kernel/config/usb_gadget/
mkdir -p midi_over_usb
cd midi_over_usb echo 0x1d6b > idVendor # Linux Foundation
echo 0x0104 > idProduct # Multifunction Composite Gadget
echo 0x0100 > bcdDevice # v1.0.0
echo 0x0200 > bcdUSB # USB2 mkdir -p strings/0x409
echo "fedcba9876543210" > strings/0x409/serialnumber
echo "Your Name" > strings/0x409/manufacturer
echo "MIDI USB Device" > strings/0x409/product
ls /sys/class/udc > UDC
```

Il faut éditer l'intérieur des guillemets des lignes 'manufacturer' et 'product' avec ceux du produit MIDI en question.

Après avoir sauvé ce fichier, il faut faire en sorte qu'il s'exécute au démarrage de la Raspberry. Pour ce faire, il faut éditer le fichier `/etc/rc.local` et rajouter avant « `exit0` » la ligne suivante :

```
/usr/bin/midi_over_usb
```

Il reste à reboot la Raspberry et effectuer la commande `amidi -l` qui permet de lister les ports MIDI disponibles. Le résultat devrait ressembler à ceci :

```
Dir Device Name
IO hw:0,0 f_midi
```

Installation des Librairies Python

```
pip install mido
sudo apt-get install libasound2-dev
sudo apt-get install libjack-dev
pip install python-rtmidi
```

Voici donc toutes les librairies utiles pour l'utilisation du port MIDI avec Python, mido étant la principale et nécessitant le reste des 3 librairies qui suivent.
Pour savoir quel objet MIDI est connecté et lequel nous voulons choisir, il suffit de lancer python en ligne de commande et d'utiliser les méthodes suivantes :

```
>>>import mido
>>>mido.get_output_names()
```

Cette commande affichera le nom du matériel connecté sous la forme `f_midi:f_midi 16:0`. Ce nom sera retranscrit dans le début du code python, dans la fonction ci-dessous :

```
port = mido.open_ input('f_midi')
```

Cette variable sera utile tout le long de notre code.

Si l'installation a été faite correctement, vous allez pouvoir tester votre piano.
Les étapes à suivre sont :

1. Connectez votre piano si ce n'est pas fait
2. Connectez un baffle à l'entrée jack de la Raspberry Pi
3. Placez-vous dans le dossier RobotProject
4. Lancez python3
5. Importez le fichier via **`from MIDIControl import midi_test as p`**
6. En exécutant la commande suivante **`print(p.play_piano())`**

Après la 5ème étape, vous devriez entendre des sons de piano lorsque vous appuyez sur ses touches. Ces touches seront imprimées sur votre écran. Après 7 secondes d'inactivité, la séquence que vous avez jouée sera affichées.

Si vous n'entendez rien, il se peut que la sortie son de votre Raspberry soit mal configurée. Vous pourrez changer cela via **sudo raspi-config**.

4.9 MODIFICATION DES MUSIQUES ET DES NOTES DU PIANO

Deux choses peuvent être modifiées à votre guise pour coller à vos besoins et envies : les notes jouées par le piano ainsi que les musiques à jouer.

Pour la première chose, le dossier *notes* contient toutes les notes nécessaires au piano. Si vous souhaitez changer ces notes, il suffit de remplacer chaque fichier par un son que vous préférez mais il faut garder le nom qu'ils ont.

Pour la seconde chose, il vous est possible d'ajouter des musiques à jouer à votre guise : le fichier **songs.txt** permet d'ajouter le chemin des musiques. Par exemple, une musique déjà présentes sur la Raspberry Pi se trouve dans le dossier **MIDIControl/songs** sous le nom de **tequila.wav**. Si vous voulez rajouter une musique, ajoutez simplement le chemin à la suite. Il faut également ajouter la séquence de note correspondante.

4.10 SYNTHÈSE VOCALE

La synthèse vocale utilisé pour les explications utilise la librairie festival. Celle-ci est basé sur un tts anglais donc les phrases ne peuvent être dans une autre langue.

Pour tester si les commandes vocales fonctionnent, suivez ces étapes :

1. Connectez un baffle si ce n'est pas encore fait
2. Placez-vous dans le dossier RobotProject
3. Lancez python3
4. Importez le fichier via **`from textToSpeech import speak`**
5. Utilisez la fonction speak comme ceci : **`speak('phrase de test')`**

Une fois la 5ème étape effectuée, vous devriez entre une voix prononçant votre phrase.

Comme pour la partie MIDI, si vous n'entendez rien, regardez si votre sortie son est correctement configurée.

4.11 ENVOI DES DONNÉES ENTRE RASPBERRY PI ET ARDUINO ET TEST DES POMPES

Pour tester les pompes et l'envoie des données entre Raspberry Pi et Arduino par la même occasion, nous allons utiliser le fichier `cocktail_melo.py` du dossier cocktail_melo.

Procédez de la façon suivante :

1. Commencez par uploader le code Arduino fourni dans l'Arduino utilisé
2. Connectez les pompes à l'Arduino et au driver
3. Connectez l'Arduino à la Raspberry Pi
4. Placez-vous dans le dossier RobotProject
5. Lancez Python3
6. Importez le code via **`from cocktail_melo import cocktail_melo`**
7. Envoyez une séquence via **`sendCocktail(nr)`** où nr vaut entre 0 et la longueur de la liste de séquence

Après cette dernière étape, les pompes devraient commencer à tourner. Si vous mettez une autre valeur pour nr, les pompes devraient tourner temps différents à chaque fois.

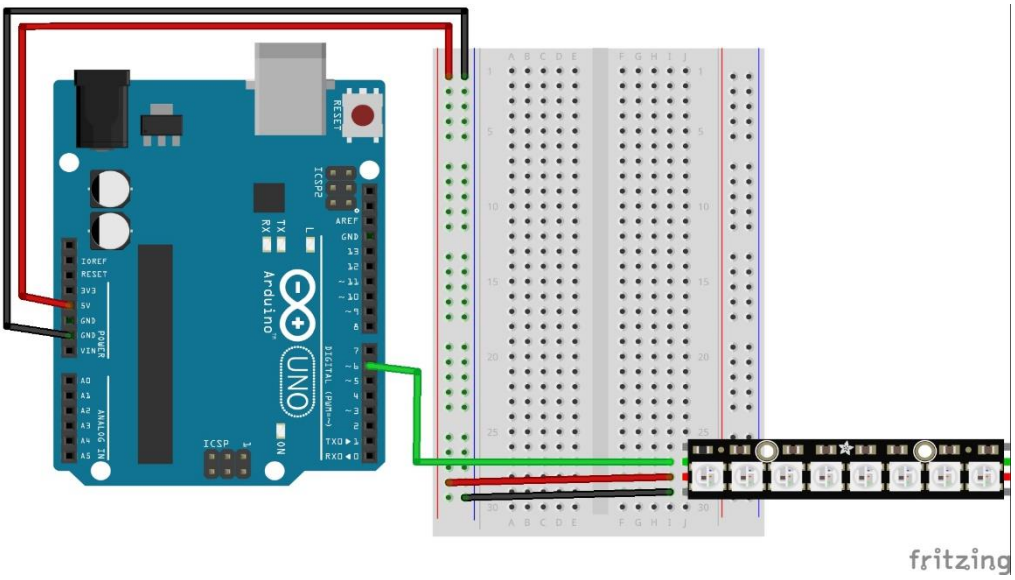
Si vos pompes ne tournent pas, essayez ces différentes solutions :

- Parfois l'Arduino peut changer de port et prendre un port différent de celui dans les codes, souvent quand vous le débranchez et rebranchez de la raspberry. Il suffit juste de reboot.
- Changez grâce aux potentiomètres sur les drivers l'entrée en courant, il se peut que parfois le courant ne soit pas suffisamment important.

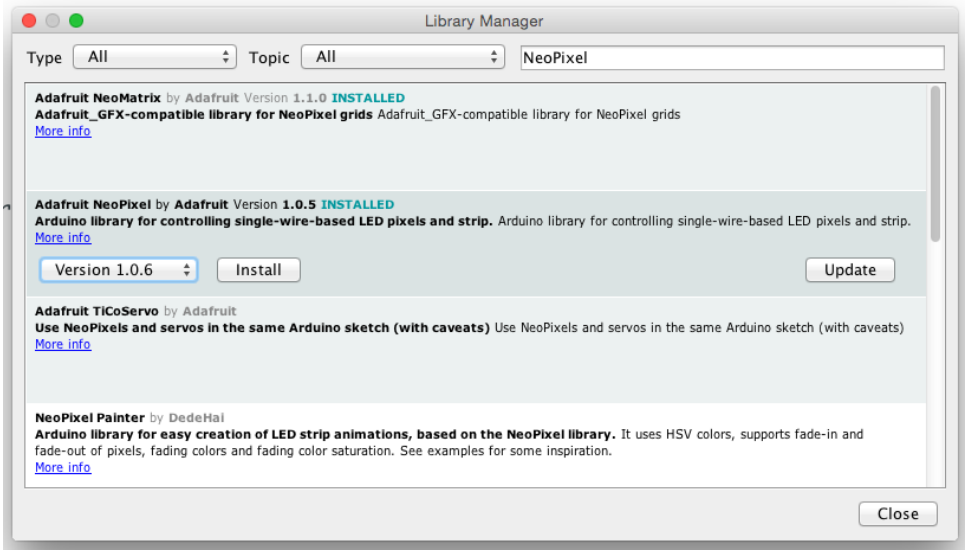
4.12 TEST DE LA BANDE LED

La bande LED est la Neopixel d'Adafruit « NeoPixel Digital RGB LED Strip - Blanc 30 LED – BLANC ». On la place sur le piano comme on peut le voir sur l'image ci-dessus.

Le néopixel a pour fonction d'aider notre pianiste amateur à jouer la mélodie qu'il a choisi. Pour chaque note une led du neopixel, qui correspondra à une touche du piano, s'allumera. Notre pianiste saura donc quelle touche du piano il devra jouer pour avoir son cocktail 😊.



Faudra connecter les pins 5V et GND de l'Arduino aux pins +5V et GND du neopixel respective, par contre l'entrée de données Din du neopixel peut provenir de n'importe quelle broche numérique sur l'Arduino.



Concernant le code pour contrôler le neopixel, il faudra installer la bibliothèque Adafruit_NeoPixel via le gestionnaire de bibliothèque de l'interface Arduino ou manuellement via ce lien https://github.com/adafruit/Adafruit_NeoPixel/archive/master.zip. Pour ce projet nous avons utilisé la version 1.6.2 qui était la version la plus récente.

4.13 EXÉCUTION DU PROGRAMME ET LANCEMENT AUTOMATIQUE

Pour lancer le programme à partir du dossier pi, il suffit simplement de taper la commande suivante en ligne de commande :

```
python3 RobotProject/main.py -c -b -p -i -a
```

Les arguments que suive la commande représente les éléments du piano à activer ou désactiver. Par exemple, si vous souhaitez retirer la reconnaissance d'âge, il suffit simplement de retirer l'argument '-c'. Pour savoir ce qu'active les autres arguments, il vous suffit de taper la commande :

```
python3 /RobotProject/main.py --help
```

Une liste de tous les arguments possibles à mettre sera affiché et chaque argument sera expliqué.
Étant donné que le programme possède une boucle infinie, le programme ne s'arrêtera jamais sauf en cas d'erreur. Si vous souhaitez l'arrêter manuellement, un simple « CTRL + C » arrêtera le programme.

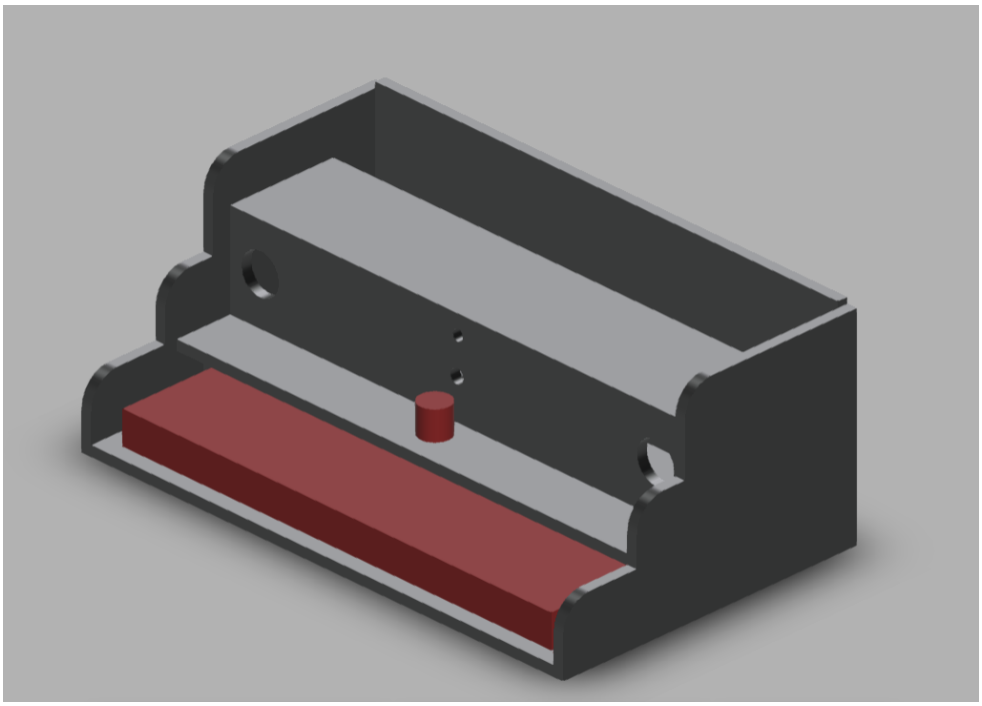
(OPTIONNEL) Si vous voulez que votre projet soit accessible sans avoir besoin de se connecter à la Raspberry, il est possible de lancer le programme au démarrage de la carte. Pour ce faire, accédez au fichier 'profile' via cette commande (si le fichier ouvert est vide, mettez-vous à la racine de la Raspberry) :

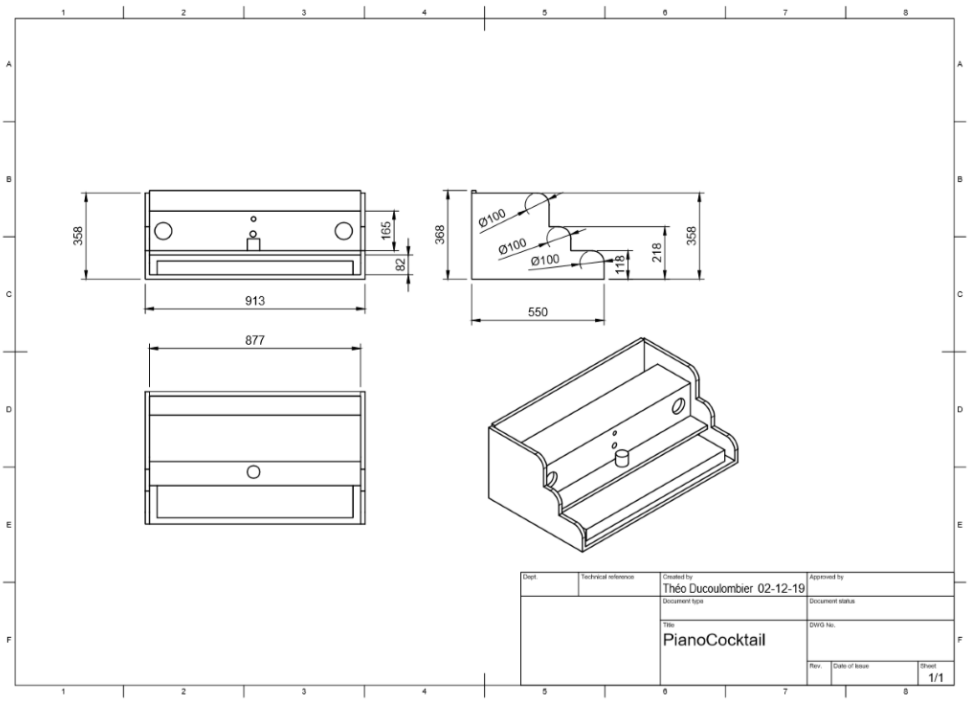
```
sudo nano /etc/profile
```

À la fin de ce fichier, indiquez le chemin du main.py avec la commande python3, suivi de tous les arguments que vous souhaitez. Par exemple :

```
python3 /home/pi/RobotProject/main.py -c -b -p -i -a
```

4.14 ASSEMBLAGE DE LA BOITE





Voici les plans utilisés pour faire la structure du piano. La fixation des planches a été fait avec des vis et des tasseaux. Les tasseaux ont été coupés au fur et à mesure pour convenir au mieux dans l'assemblage des planches et avoir un ensemble solide. De plus, pour la charnière permettant de fixer la planche arrière, il est conseillé de la surélevé en mettant les charnières sur un tasseaux permettant de plus facilement passer des câbles dans le fond. La planche du dessus permet un espace avec la planche du fond. Cette planche peut être donc raccourcis à votre guise en fonction de la taille de vos bouteilles. Les trous des haut-parleurs sont à faire en fonction de vos modèles d'haut-parleurs.

4.15 JOUEZ !

Voilà ! À vous de jouer !

Le système, lors du lancement, demandera de déposer votre verre sur le bouton pour lancer la suite du programme. Une fois que ce sera fait, une brève explication sera lancée pour vous montrer comment sélectionner la musique que vous souhaitez jouer. La première et troisième touche vous permettra de naviguer entre les différentes musiques disponibles et la deuxième touche permettra de la sélectionner. Ensuite, vous pourrez jouer. Les touches que vous devez enfoncer pour reproduire la musique seront illuminés par la bande LED. Dès que vous aurez fini et que la séquence jouée est correcte, le robot vous servira votre cocktail.