Smart Armoire

# Objectifs

Presentation 15min, rapport succint/pour pouvoir reprendre le travail

Dans le cadre de notre dernier projet à l’INSA, nous avons choisi de travailler sur un projet cumulant reconnaissance vocale et maquette pilotée : l’armoire capricieuse.

L’objectif du projet est de commander l’ouverture d’une armoire par la voix avec la contrainte de devoir répondre à un questionnaire pour pouvoir déverrouiller l’armoire.

L’objectif de ce rapport est de présenter le travail effectué et de permettre une poursuite /amélioration du projet par les futurs étudiants de l’INSA

Nous avons pour obligation d’utiliser une Jetson Nano (carte Nvidia), nous pouvions passer des commandes et utiliser le matériel mis à disposition par l’INSA.

# Ancien Projet

Le sujet avait déjà été donné l’année dernière mais celui-ci n’avait pas été aboutis. Nous avons eu accès à leurs délivrables.

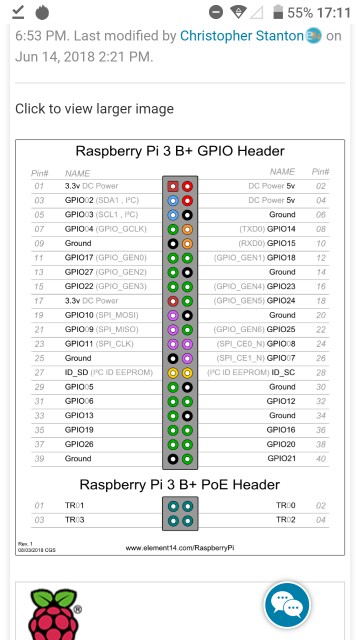
Quand nous avions récupéré le projet, nous avions une carte son nvidia pour la jetsonNano sans la partie code car ils n’avaient pas réussi à l’utiliser ; nous avions une structure en profilés avec des plexiglas et une boite imprimée en 3D.

La structure n’était pas très bien montée, ce qui causait le fait que la boite ne rentrait pas dans les 4 emplacements. Les plexiglas avaient été découpé à la scie sauteuse, ce qui causait une irrégularité dans la coupe, ils avaient également rencontré des difficultés de perçage, rendant l’état des plexiglas de mauvaise qualité.

Concernant la partie hardware, ils avaient fait un choix de composants. Leurs choix pour les voyants et bouton étaient des éléments d’automates, ils devaient être piloter en 12V. Afin de pouvoir les contrôler, ils passaient par des relais. Malheureusement ils étaient mal dimensionnés, le courant appliqués les a faits chauffés et donc ils étaient inutilisables quand nous les avons pris.

# Matériel

A voir



# Le projet

## Définition des fonctionnalités de l’armoire

A faire

## Choix et commande du matériel

A faire

## La structure

La première étape é été de démonter et remonter entièrement la structure, le but était de savoir nous pouvions la réutiliser. Nous avons donc constaté que les profilés était bon et en les montants correctement nous arrivons à une structure droite et stable. Nous avons décidé de récupérer la structure existante mais de la modifier légèrement, notamment en ajoutant des portes à l’armoire et en recoupant les plexis entièrement.

L’idée originale était de ne pas consacrer trop de temps à cette partie et de pouvoir avancer sur le reste. Nous nous sommes aperçus par la suite que cela était une erreur car lors de la construction de l’armoire et de la mise en place de notre système d’ouverture, il a été difficile de le faire car non pensé dès le départ.

Une image contenant jaune, plancher, intérieur, meubles

Description générée automatiquement

En ce qui concerne les parois en plexiglas, notre plan initial était de pouvoir utiliser la découpeuse jet d’eau de la plateforme mécanique de l’INSA. Pour cela des plan 3D des plexiglas avait été réalisé, malheureusement à la suite d’une erreur de notre part concernant des problèmes de dimensions sur les plans, les plexiglas découpés était complètement inutilisable. Pour donner suite à cela nous avons refait des plan 3D sur Creo aux bonnes dimensions et les avons utilisés afin de créer des plans 2D. A cause de la pause des vacances de noël nous avons donc dû découper nous même les plexiglas. Cela a été réalisé à l’aide d’une scie à ruban personnelle.

Nous avons vitré tous les côtés sauf ceux de la face avant afin de simplifier le passage des câbles et la fixation des moteurs. La plaque du dessus est glissée dans les profilés et bloquée par des vis. La raison de ce choix est de pouvoir la retirer rapidement. De plus, si l’on veut que la jetson soit dans l’armoire en étant sur le support, il faut la retirer car l’ensemble est trop haut.

## L’ouverture des portes

Pour l’ouverture des portes, nous avons décidé d’utiliser un système de poulie et de glissière. La porte sera donc fixée à une glissière et à un fil continue et entrainée par un moteur. L’armoire est composée de 2 niveaux et chacun sera composé d’une porte. L’idée est que la porte coulisse d’un côté pour pouvoir ouvrir une seule case en même temps. Afin de pouvoir ouvrir la deuxième porte, la plaque de plexi sera translatée de l’autre côté.

Une image contenant intérieur

Description générée automatiquement

L’installation des glissières a été faite grâce à des plaques métalliques. La glissière est fixée sur cette plaque et celle-ci est fixée sur les profilés. Idéalement il faudrait refaire la structure pour pouvoir fixer les glissières directement dans les rails, de même pour les poulies et le moteur. Afin de réaliser cette plaque, nous avons donc découpé une chute d’une tige que nous avons percé selon l’écartement des vis des glissières.

Une image contenant texte, jaune, transport, école

Description générée automatiquementUne image contenant intérieur, rayon

Description générée automatiquement

Les glissières sont positionnées une fois en-dessous et une fois au-dessus des cases de rangements. Pour pouvoir positionner la glissière inférieure, nous avons dû procéder à un rajout sur la structure. La partie inférieure sert donc à pouvoir positionner les glissières.

## Fixation du moteur et des capteurs

## Utiliser des moteurs avec une Jetson nano

Lorsque l’on lit la datasheet de la jetson nano, il n’y a pas de sortie PWM (Pulse with modulation) d’indiquée. Or une telle sortie est nécessaire afin de pouvoir contrôler un moteur.

Les Jetson nano utilisent le même schéma PCB que des raspberry Pi. Il y a donc 4 pins PWM sur la Jetson mais ils ne sont pas configurés de base. Les pins qui peuvent être configurés en PWM sont le 12, 32, 33, 35.

### Configurer la sortie des pins en PWM de la carte

La première étape est de configurer les pins de la Jetson. Le gitub indiquant les instructions se trouve en annexe. Pour cela il faut dans le terminal de la Jetson exécuter la commande suivante :

sudo /opt/nvidia/jetson-io/jetson-io.py

Cela ouvre les paramètres de configuration de la carte, il faut suivre les étapes suivantes :

1. Sélectionner : Configure Jetson 40pin Header
2. Sélectionner : Configure Header pins mannually.
3. Choisir dans le menu les pin 32 et 33 est les passer en PWM
4. Enregistrer
5. Quitter
6. Rebooter la Jetson nano

Tous les autres fonction et programme sont fait dans le docker\_file qui est notre environnement virtuel. Le nôtre s’appelle SmartArmoire, il est disponible sur github.

### Utiliser les sortie GPIO dans le code

La première chose à faire est d’appeler la bibliothèque GPIO. Dans le code il faut déclarer que les sorties sont en PWM. Pour cela on définit le type de board que l’on a avec GPIO.setmode(GPIO.BOARD), le programme reconnait automatiquement le type de carte et s’adapte en fonction. Dans notre programme les sorties utilisées sont celles des GPIO32 et GPIO33.

Une image contenant texte

Description générée automatiquement

## Utiliser la carte son de la Jetson nano

Nous avons utilisé la carte son USB pour Jetson Nano (référence Waveshare : WV19491). Cette carte comprend un micro des hautparleurs et de quoi régler le son.

### Initialisation carte

La carte est connectée par USB à la jetson nano. La première étape est de voir si elle est reconnue. Dans les paramètres son de la jetson, il faut bien sélectionner la carte comme périphérique par défaut.

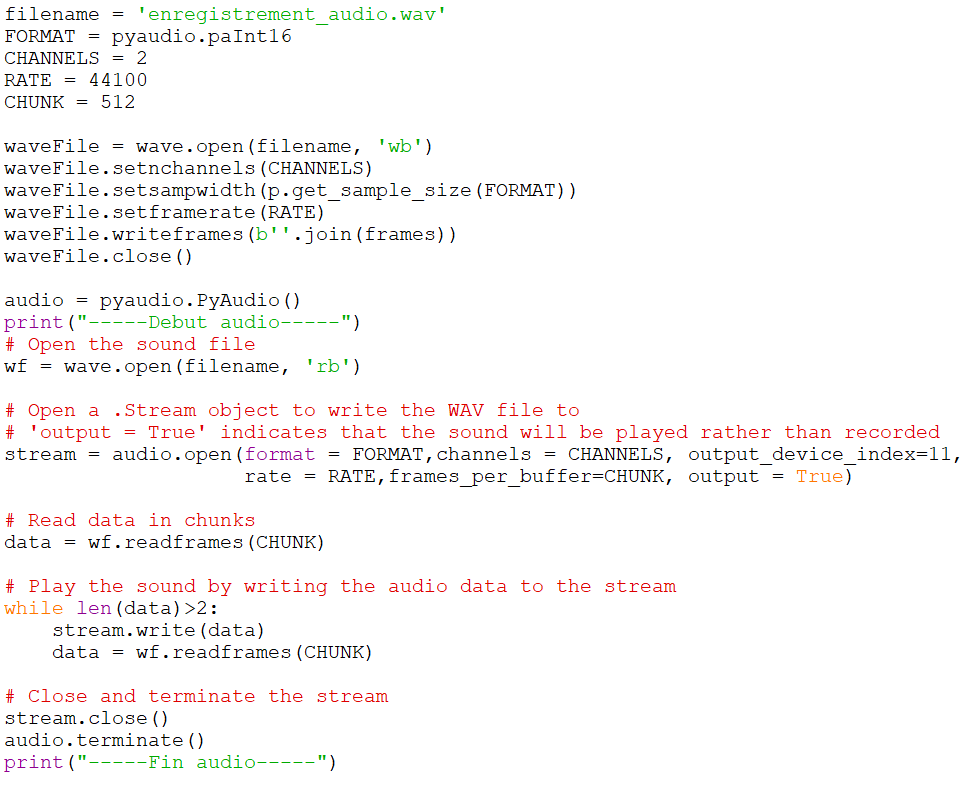
Il faut également vérifier le nombre de channel associé à cette carte et l’index de l’output. Pour cela il suffit de lancer le script Test\_son.py. Ce script permet d’afficher toutes les entrées/sortie audio avec le nombre de channel d’entrée et de sortie. La carte son est nommée avec le nom « USB PnP Audio Device », ici l’index est le 11.

Une image contenant texte

Description générée automatiquement

### Emettre en audio

La première étape est de sélectionner le fichier audio à émettre. La deuxième étape est de le transformer pour s’assurer que le format soit le bon. A bibliothèque utilisée est PyAudio. Nous lisons par la suite le fichier avec comme paramètre le nombre de frame et la fréquence à utiliser. Il faut bien indiquer l’utilisation de 2 channel de sortie car nous sommes en stéréo et de l’utilisation de la sortie 11 (output\_device\_index).



### Enregistrer avec le micro

Pour enregistrer un fichier audio nous utilisons la bibliothèque PyAudio. La fonction enregistrement permet d’enregistrer dans un fichier que nous avons nommée « enregistrement\_audio ». Il s’agit d’un fichier wav. Nous enregistrons toujours avec le même nom de fichier donc il n’y a pas de sauvegarde des enregistrements.

Une image contenant texte

Description générée automatiquement

Nous avions essayé plusieurs autres bibliothèques mais malheureusement nous avons rencontré des difficultés à cause de la carte son, en effet celle-ci n’était pas reconnu par les autres bibliothèques et donc la mauvaise entrée audio était utilisée.

## Reconnaissance vocale

Pour la reconnaissance vocale nous avons décidé d’utiliser la bibliothèque SpeechRecognition. Il s’agit d’une bibliothèque qui nécessite internet. Nous avons choisi de séparer la reconnaissance vocale de l’enregistrement. La reconnaissance se fait donc à partir d’un fichier audio préalablement enregistré. Nous avons constaté qu’il fallait que la connexion internet devait ce faire depuis un partage de connexion qui ne provient pas d’un Iphone. La bibliothèque fonctionne de manière beaucoup plus fluide depuis un ordinateur, cela fonctionne sur la Jetson mais cela est plus lent, d’autant plus depuis un partage de connexion dans la salle. Cela donne un peu de latence involontaire au système.

Une image contenant texte

Description générée automatiquement

Une solution possible serait le fait d’utiliser une bibliothèque qui ne recours pas à internet, tel que sphinx.

# Conclusion