Hector BELAUBRE
Romain CORRADINO
Guillaume DUPUIS
Hamed HAOUES
Gaël ZAVALETA

Groupe 1.2

# Rapport : Voice Control of an Unmanned Vehicle

Encadré par Monsieur M'Sirdi

Année 2018-2019

Polytech Marseille en collaboration avec Frankfurt University of Applied Sciences

Sommaire :



Introduction :

Nous étudions dans ce projet le contrôle d'un drone à l'aide de la voix. Nous nous sommes posé un certain nombre de questions car le sujet est vaste. Pour commencer, nous avons essayé de synthétiser notre sujet de projet. Le contrôle du drone par la voix : il faut avant tout que ce soit intuitif afin de pouvoir utiliser notre système facilement. Pour cela, on utilisera donc un smartphone et plus précisément une application que nous avons développé. Il faut ensuite transmettre l'ordre donné à l'application, au système. Dans ce but, on va utiliser un microprocesseur ainsi qu'un module Bluetooth. On enverra alors les ordres au système par cet intermédiaire qui les traitera puis qui demandera à notre drone de les exécuter.
Notre projet se découpe donc en trois parties distinctes : la reconnaissance vocale, la transmission et enfin le traitement de l'information reçue ainsi que l'exécution de l'instruction demandée par l'utilisateur.
Mais avant toute chose bien que notre projet ne porte pas sur l'histoire du drone, il est important de savoir de quoi on parle. C'est pourquoi dès le début de notre projet nous avons réalisé des recherches sur l'histoire du drone, les différents types de drones qui existent, les fonctions que certains possèdent…

Partie I : Histoire du drone

Bien que notre projet ne porte pas sur l'histoire du drone, il est important de savoir de quoi on parle. C'est pourquoi dès le début de notre projet nous avons réalisé des recherches sur l'histoire drone, les différents types de drones qui existent, les fonctions que certains drones possèdent…
Tout d'abord qu'est ce qu'un drone ? Un drone c'est avant toute chose un véhicule sans équipage.
Le drone était à son origine considéré comme un véhicule aérien sans pilote à l'intérieur. Les recherches sur le drone ont commencé il y a une centaine d'années et notamment en France, Georges Clémenceau, le président de la République de l'époque, lança des recherches pour développer des avions sans pilote à la fin de la Première Guerre mondiale. En même temps, aux Etats-Unis, un projet similaire voit le jour. Les premiers prototypes ont vu le jour au début des années 1920 mais l'armée notamment ne voyait pas vraiment d'intérêt au développement de ces véhicules. Ces premiers modèles de drones fonctionnaient par radio-télécommandes et utilisaient des moteurs d'avions classiques.
Bien qu'initialement le but du drone n'était pas très clair, on se rendit compte qu'il pouvait s'agir là de très puissante armes. Et dès la Seconde Guerre mondiale le drone est utilisé afin de bombarder les forces ennemis des deux côtés. Mais le grand essor du drone date de la Guerre de Corée et de la Guerre du Viêt-Nam durant lesquelles le drone est extrêmement développé en particulier par les forces américaines. Le drone devient alors à la fois un véhicule capable de larguer des bombes mais également un moyen stratégique très puissant capable de surveiller les ennemis sans mettre en danger des vies humaines. Le drone qui était déjà un puissant engin d'intervention à l'aide de largage de bombe devient alors aussi un instrument de surveillance sophistiqué contre lequel il est difficile de lutter. Voici un drone utilisé lors de la guerre du Viêt-Nam dans les années 1960 :

Depuis cette époque le drone s'est encore beaucoup développé. Il est aujourd'hui très régulièrement utilisé lors d'intervention à distance comme par exemple au Moyen-Orient où la Communauté Internationale visent et repèrent les cachettes des groupes terroristes à l'aide de drone très sophistiqués possédant plusieurs caméras hautes définitions, une autonomie allant jusqu'à 36 heures etc… Le système de transmission des drones militaires actuels a également évolué et ne sont plus radio-télécommandé mais cette transmission s'effectue par télécommunication, soit en portée optique (le système utilisé par les avions pour communiquer avec le sol) sur de courtes distances (jusqu'à 150 km), soit en utilisant un relais, ce dernier pouvant être un satellite ou un autre vecteur aérien (c'est à dire un avion ou un autre drone).

Le drone s'est aujourd'hui énormément démocratisé, leurs utilisations n'est plus réservé à un usage militaire et ils peuvent aussi être utilisés comme objet de loisir ou comme un outil aidant à la réalisation d'oeuvres culturelles (Le clip d'Orelsan "Basique" a été entièrement tourné à l'aide d'un drone dans un plan-séquence de 3 minutes). Aujourd'hui les drones sont également de plus en plus accessible à tous. En effet des entreprises comme Exolys, Midrone, Parrot ou encore DJI propose des drones pour tout les budgets (de 50€ à 1300€).

La définition original de drone qui était considéré uniquement comme un véhicule aérien auparavant s'est élargi aujourd'hui car on nomme drone tout véhicule terrestre, marin ou aérien sans pilote à bord.

Aujourd'hui le drone a donc plusieurs fonctions. Certains servent à un usage militaire alors que d'autres sont spécialisés dans la prise de vue. Par exemple il existe le drone "selfie". Cet engin peut prendre un selfie de son utilisateur. Il possède plusieurs modes comme le mode "orbit" : avec ce mode, le drone tourne autour de vous durant la prise de vue afin de prendre une courte vidéo ou simplement de prendre une photo "selfie" en face de vous.
D'autres drones ont comme fonctionnalités de suivre l'utilisateur pour capturer un exploit sportif par exemple. On l'appelle aussi le "dronie". Le mode "Follow me" qui permet donc de suivre l'utilisateur fonctionne grâce à un Face tracking (suivi du visage).
Un nouveau type de drone pourrait également bientôt arriver : ce sont les drones livreurs à domicile ! Ce projet lançait par Amazon permettrait à l'entreprise de faciliter et d'accélérer les livraisons des commandes à domicile. Cependant le projet bat un peu de l'aile car cela coûterait

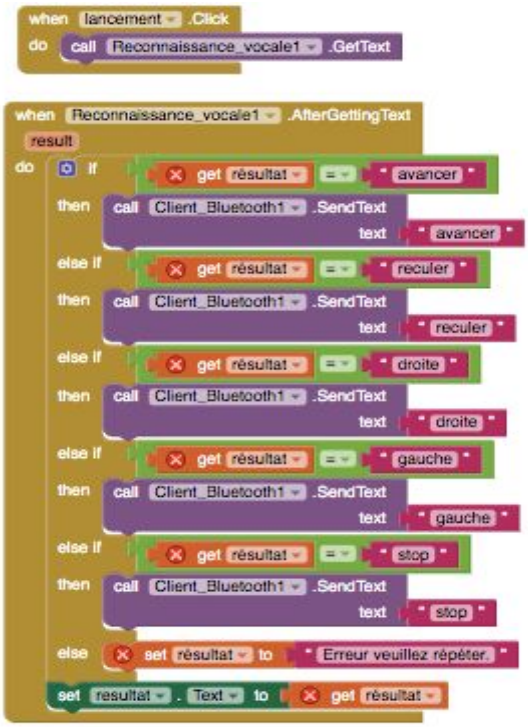
énormément d'argent et les drones ne pourraient porter qu'une charge limité. De plus la réception du colis chez le client est un autre problème. Que faire si personne n'est présent pour réceptionner le colis ? Amazon a également lancé en parallèle le développement d'un autre drone mais cette fois ci il s'agit d'un drone terrestre. Ce robot livreur de colis est actuellement testé dans le comté de Snohomish, dans la banlieue nord de Seattle. Ces robots sont des chariots à 6 roues. Si ces nouveaux moyens de livraisons continue de se développer et de montrer des résultats convaincants, cela permettra dans un futur proche de réduire considérablement la pollution liée au camion de livraison. En même temps cela créera des emplois pour vérifier le bon fonctionnement de ce système de robots intelligents.
Pour tout ce qui est des drone militaire, le système de commande est extrêmement perfectionné et les drones volent avec un pilote automatique et l'Homme n'intervient que sur la position GPS à laquel on souhaite que le drone aille, l'altitude que l'on désire etc…

Photo d'un drone de guerre utilisé actuellement :

Photo d'un drone que l'on peut trouver actuellement dans le commerce :

Partie II : Reconnaissance vocale

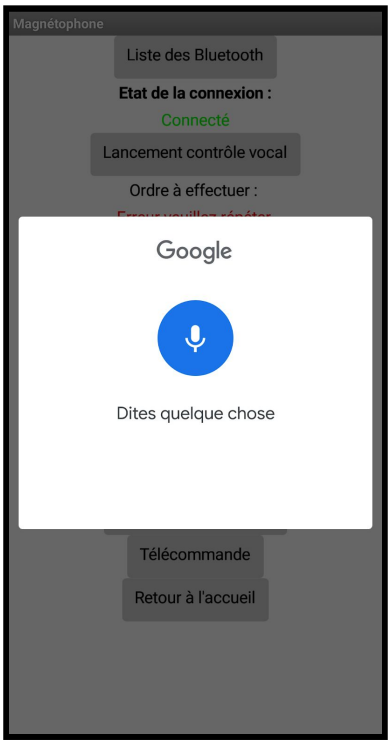
Dans cette partie on va s'occuper de tout ce qui concerne la reconnaissance vocale.
Google a développé une application web : MIT App Inventor dont on va s'aider pour créer notre propre application. Le but est simple : mettre en place une reconnaissance vocale infaillible !
Pour cela le logiciel nous propose toute sorte de plug-in (c'est à dire de petit logiciel que l'on va rajouter pour compléter, ajouter de nouvelles fonctionnalitées).
Un de ces plug-in se nomme "Reconnaissance vocal". Super ! Mais ce n'est pas du tout fini. MIT App Inventor fonctionne avec un système de "blocs". On doit agencer des blocs entre eux pour que notre application fonctionne. Voici une partie de ces "blocs" avec des explications associées :




Lorsqu'on appuie sur le bouton "Lancement reconnaissance vocale", un enregistrement se lance.

Une fois l'enregistrement réalisé, cet enregistrement est analysé par la reconnaissance vocale qui le retranscrit à l'écrit ensuite. Si l'ordre donné correspond à l'un des différents ordres enregistrés (avancer / reculer / droite / gauche / stop) alors la suite du processus se met en route (transmission vers le microprocesseur) sinon un message d'erreur est affiché (cf captures d'écran suivante).
(Les blocs violets "call Client_Bluetooth…" seront étudiés dans la deuxième partie).

Voici des captures d'écran sur l'application de ce qui correspond aux blocs précédent :




Lancement de la reconnaissance vocale

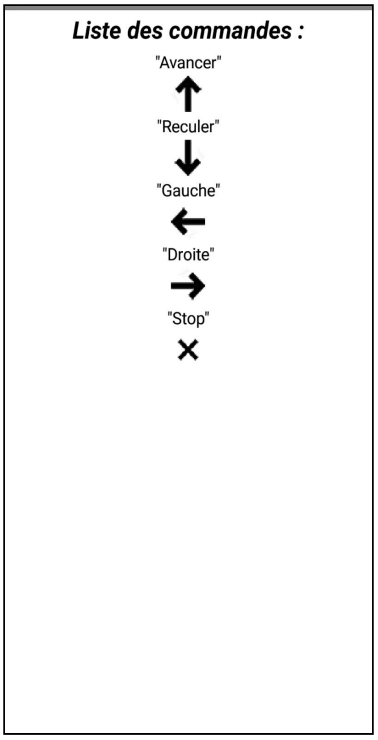

Ordre donné faisant parti de la liste de consignes à exécuter


Ordre donné ne faisant pas parti de la liste de consignes à exécuter

On va maintenant s'intéresser aux autres menus que nous avons incorporé à l'application pour aider l'utilisateur à utiliser le système ainsi que pour lui donner d'autres options pour diriger la voiture.

Le bouton "Liste des commandes" donne les différentes options possibles (avancer, reculer etc…).
Le bouton "Télécommande" propose une télécommande de base si on souhaite utiliser l'application comme une simple télécommande et n'implique pas de reconnaissance vocale mais simplement une alternative si on ne souhaite pas utiliser la reconnaissance vocale pour contrôler le drone.
Enfin le bouton "Retour à l'accueil" propose simplement de retourner au menu.

Voici des captures d'écrans du menu de notre application :




Un petit menu simple donne les ordres à donner à l'application pour faire fonctionner le drone.

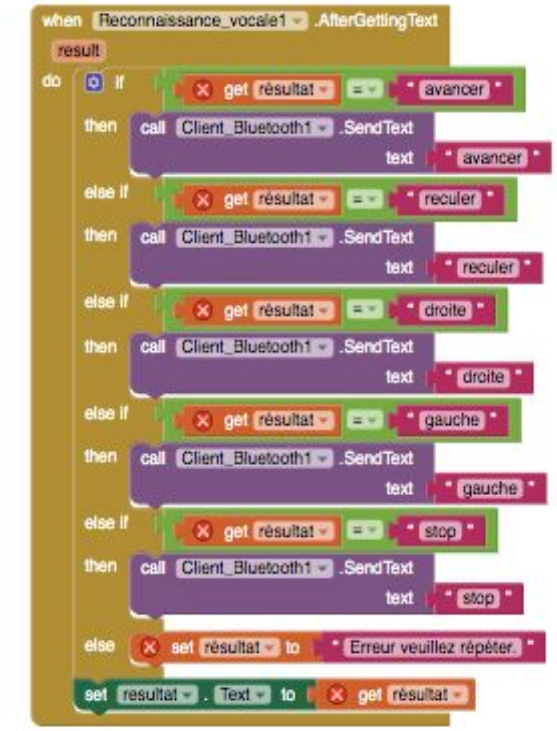

Si l'utilisateur souhaite contrôler son drone à l'aide d'une télécommande classique, l'application le propose.


Ce menu simple donne quelques informations sur notre projet et le bouton "Magnétophone" renvoie directement vers la page vue précédemment.

Conclusion de la Partie II : Notre application est maintenant capable de capturer notre voix, de l'analyser et de déterminer ce qu'on a dit. Si ce qu'on a dit ne correspond pas à l'un des ordres préenregistrés ou que l'application n'a pas bien capté les paroles, l'application demande alors à l'utilisateur de répéter.

Partie III : Transmission entre l'application et le système

Notre application capte donc maintenant nos paroles. Mais comment les transmettre à notre microprocesseur ? Et sous quelle forme ?
C'est ce que nous allons voir maintenant :
Tout d'abord nous avons décidé de choisir un module Bluetooth pour notre système car nous avons pu en trouver un facilement et que cela nous assure de pouvoir continuer notre projet sans attendre de recevoir un module wifi qui ne serait peut-être pas arriver à temps. De plus MIT App Inventor permet de créer des liaisons en Bluetooth avec d'autres appareils de façon assez simple. Le choix du Bluetooth nous est donc parvenu comme une évidence.
Pour commencer : comment créer le lien entre notre application et le module Bluetooth Arduino ?
Nous allons pour cela, dans un premier temps, demander à l'utilisateur de choisir le réseaux Bluetooth où se connecter (dans notre exemple le nom du réseau correspond à celui d'un ordinateur et non celui du module Bluetooth). L'application se connecte alors au module Bluetooth. On voit par ailleurs que l'état de la connexion a changé. Cela permet à l'utilisateur de vérifier à tout moment qu'il est toujours connecté à son microprocesseur. Une fois la connexion réalisée , dès lors que l'utilisateur donnera un ordre correct au système, celui-ci sera transmis par Bluetooth. Comment ? à l'aide des blocs violet commençant par "call Client_Bluetooth…" :



Voyons un exemple précis d'un peu plus prêt pour bien comprendre comment fonctionne le processus : (la connexion Bluetooth est réalisée)

Tout d'abord on lance l'enregistrement de l'ordre de l'utilisateur :


Puis une fois l'ordre donné et l'enregistrement terminé, l'application comprend l'ordre demandé :


Dans notre exemple supposons que l'utilisateur demande au drone d'avancer, alors si le "résultat" (ordre donné par l'utilisateur et retranscrit à l'application à l'aide de la reconnaissance vocale) est "égale à" (c'est à dire écrit de la même manière) à "avancer" ALORS (then en anglais) on appel le Client Bluetooth c'est à dire notre module Bluetooth Arduino en lui disant "Je vais t'envoyer des informations". Ces informations correspondent tout simplement au mot "avancer". Mais alors on peut envoyer des mots à Arduino ? On n'est pas censé communiquer en binaire ou plus généralement avec des chiffres ? Et bien non on peut certe communiquer en envoyant des nombres qui seront converti en binaire puis reconverti mais on peut également directement demander à notre application d'envoyer des mots au microprocesseur (qui seront certe convertis dans un premier temps en binaire puis convertis en langage ASCII).

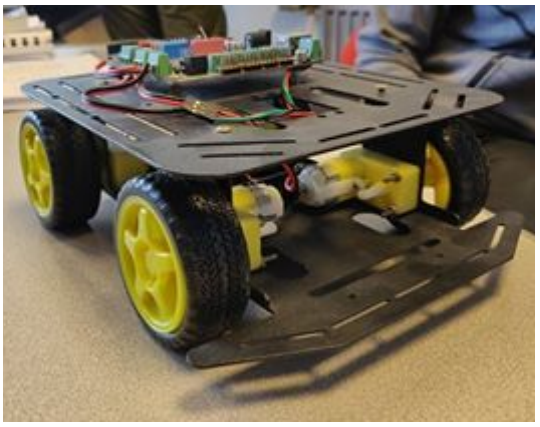
Evidemment s'il s'agit d'un message d'erreur car l'utilisateur n'a pas donné le bon ordre ou tout simplement parce que l'application n'a pas compris l'ordre à exécuter alors ce message n'est pas envoyé au microprocesseur qui ne saurait que faire de cette information dont il se moque. On peut le voir avec les blocs suivants :

Il s’agit des derniers blocs du grand “compartiment”. Ils expliquent que :
Si l’enregistrement ne correspond pas aux ordres alors la variable résultat prend désormais la valeur “Erreur veuillez répéter” et on va afficher cette valeur dans l’emplacement texte qui se situe en dessous de “Ordre à exécuter” (voir photo 3 page 2). On n’observe que comparé aux blocs précédent, celui-ci n’envoie pas de texte à notre module Bluetooth.

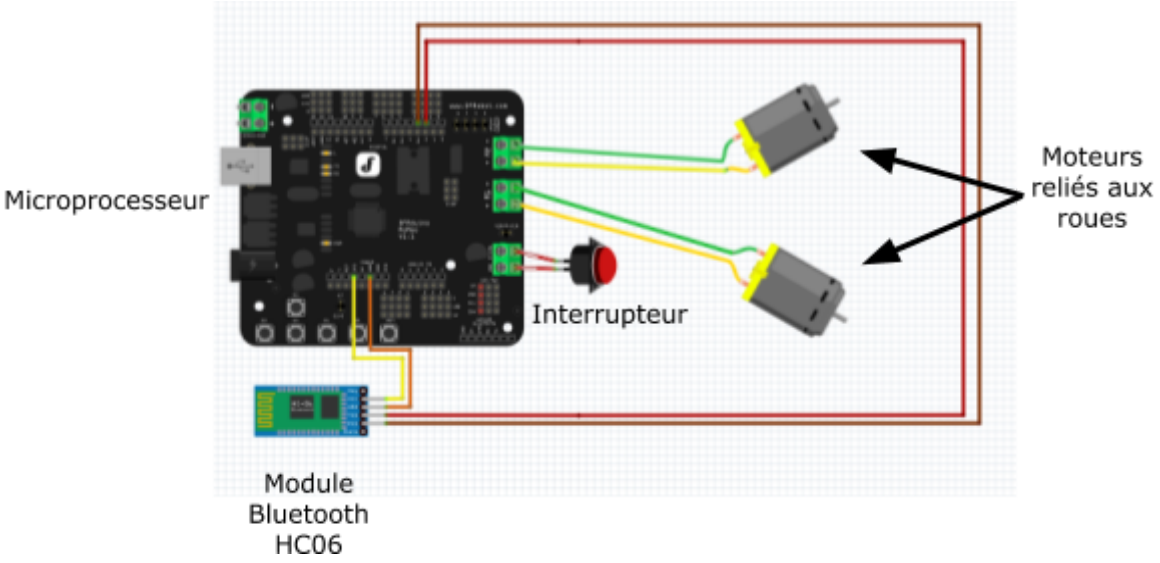
Conclusion de la Partie III :

On peut désormais à l’aide de l’application se connecter à notre module Bluetooth mais également lui envoyer du texte. Nous verrons dans la troisième partie comment on va se servir de ce texte pour faire fonctionner notre drone et lui dicter à l’aide du contrôle vocale les directions dans lesquelles il doit aller !

Partie IV : Programmation matériel et logiciel du drone :



Il a été mis à notre disposition une voiture programmable sur Arduino. Voici un schéma clair réalisé sur Fritzing de la disposition de notre drone :
On retrouve un microprocesseur dans lequel on va compiler le programme(c'est à dire enregistrer le programme dedans). On connecte les deux moteurs à courants continus sur la carte Arduino (microprocesseur) à des broches particulières (on y reviendra un peu plus tard). Ces moteurs sont connectés aux roues avants de la voiture. Il y a également un module Bluetooth HC06 connecté sur la carte Arduino(microprocesseur). Celui-ci va permettre de faire la liaison entre l'application et la voiture. Enfin on place un interrupteur pour pouvoir allumer et éteindre le drone simplement.



Le logiciel Arduino est un logiciel de programmation qui utilise le langage C++. Notre programme se découpe en trois parties :

- L'initialisation des bibliothèques et des différents modules et la définition des variables.
- La "void setup" qui correspond à la configuration des différents modules. La "void setup" n'est exécuté qu'une fois
- La "void loop" qui correspond à une boucle infini dans laquelle on donne les ordres, les conditions, les boucles que doit effectuer le système.

Tout d'abord l'initialisation des bibliothèques et des différents modules et la définition des variables :

```
#include <SoftwareSerial.h>

int pinMoteurD=5;
int pinMoteurG=6;

SoftwareSerial hc06(2,3);
String ordre;
```

On a tout d'abord "#include <SoftwareSerial.h>"
Cela permet d'inclure la bibliothèque SoftwareSerial. Cette bibliothèque est primordiale car elle permet notamment de communiquer avec l'ordinateur par USB et sans cette bibliothèque on ne pourrait compiler aucun programme de l'ordinateur vers le microprocesseur.
On a ensuite les définitions des variables pour les moteurs :

```
int pinMoteurD=5;
int pinMoteurG=6;
```

On initialise aux pins (un pin correspond à une broche dans laquelle on peut brancher un fil) 5 et 6 les deux moteurs (celui de droite au pin 5 et celui de gauche au pin 6). Le "int" indique que la variable est de type integer c'est à dire qu'il s'agit d'entier.
On initialise ensuite le module Bluetooth HC06 à l'aide de :
"SoftwareSerial hc06(2, 3);"
On place notre module Bluetooth sur les pins 2 et 3.
Enfin on nomme une variable "ordre". Cette variable est de type "String" ce qui correspond à une chaîne de caractères. C'est dans cette variable que l'on recevra l'ordre reçu par Bluetooth.

Nous venons donc d'initialiser et de définir tout ce dont on aura dans notre programme. On peut donc commencer notre "void setup" :

```
void setup(){

  Serial.begin(9600);
  hc06.begin(9600);
  pinMode(pinMoteurD, OUTPUT);
  pinMode(pinMoteurD, OUTPUT);

}
```

La "void setup" est une fonction exécutée à l'initialisation de la carte. Elle n'est exécuté qu'une seule fois au démarrage. Ici on lance tout d'abord le moniteur série qui va être initialisé à une fréquence de 9600 baud (nombre de caractère par seconde). Le moniteur série permet de vérifier que notre programme marche bien car il permet d'obtenir un retour de données du microprocesseur. Nous reverrons un peu plus tard que nous pouvons vérifier que notre drone comprend bien les ordres qu'il reçoit et les exécute.
On lance également la communication avec le module Bluetooth HC06 également avec une fréquence de 9600 baud. Enfin le pinMode(pinMoteurD, OUTPUT); indique au microprocesseur que la variable pinMoteurD qui est sur le pin 5 correspond à une sortie (OUTPUT en anglais).

Notre "void setup" étant terminé, on passe maintenant à la "void loop" :

Nous voici désormais dans la "void loop". Comme son nom l'indique il s'agit d'une boucle (loop). Afin d' éviter de copier-coller 500 fois le programme la void loop est rappelé automatiquement une fois qu'elle est terminé, de cette façon on économise de l'espace de stockage sur notre carte qui est souvent limité à quelques dizaines de kilo-octets seulement.
Il s'agit donc en quelque sorte d'une boucle infini.
On retrouve dans la void loop tout ce qui correspond aux actions que va effectuer notre système avec différentes conditions, implications etc...

Voici le programme de notre “void loop” :

```
void loop(){


  while (hc06.available())
  {

    delay(3);

    char c=(hc06.read());
    ordre += c;
    }

  if (ordre =="avancer"){
    Serial.println(ordre);
    Serial.println("La voiture avance");
    analogWrite(pinMoteurD, 200);
    analogWrite(pinMoteurG, 200);
   }
```

Tout d’abord on retrouve une condition :

“while (hc06.available())” : Ici notre condition est la suivante : Tant que le module bluetooth est disponible (c’est à dire connecter au microprocesseur) alors on continue le programme. Si le module bluetooth est débranché de la carte alors la carte comprend qu’il ne sert à rien de continuer le programme et attend que le module soit rebranché aux pins associés. Donc si le module Bluetooth n’est par exemple pas connecté aux bons pins, alors le programme ne fonctionnera pas.
Le “delay” est nécessaire car si le module n’est pas connecté alors la carte attendra trois milliseconde avant de revérifier si le module bluetooth. Cela permet d’éviter de faire chauffer le micro-processeur qui devrait alors vérifier des milliers de fois à chaque instant que le module est branché ce qui risquerait alors de le casser.

On initialise ensuite une nouvelle variable “c” qui est de type char (pour caractère). Celle ci permet de retranscrire caractère par caractère l’ordre reçu par le module Bluetooth(cette ordre correspond à hc06.read() dans le code et il s’agit de la “lecture” du modèle Bluetooth qui a reçu l’ordre provenant de l’application). Bien que l’application soit capable d’envoyer directement un mot au module Bluetooth, le microprocesseur à quant à lui beaucoup de mal à recevoir le mot directement. On place alors chaque caractère un à un de l’ordre reçu dans la variable “c”. Puis on incrémente

dans la variable “ordre” (définit dans la première partie du code) chaque lettre une à une ce qui correspond en terme de code à :
“ordre += c ;”

Une fois que tout cela est fait où en sommes nous ? Et bien cette partie du code assez complexe à comprendre correspond tout simplement à la réception de l’ordre et à l’écriture de l’ordre dans la variable “ordre”.
Que nous reste-t-il alors à faire ? Et bien nous devons maintenant, en fonction de l’ordre reçu, ordonner aux drones de les réaliser. Dans l’extrait du code plus haut, on a ce qui correspond à l’ordre “avancer”. Voyons comment cela est décrit :
L'instruction “if” correspond à une condition et exécute l'instruction ou l'ensemble d'instructions précédent si la condition est «vraie».
Notre “if” ici peut être traduit ainsi :
Si l’ordre reçu par Bluetooth est “avancer” alors :

- Dans le moniteur série écrire l’ordre correspondant (soit avancer ici)
- Dans le moniteur série encore écrire “La voiture est en train d’avancer” pour vérifier que l’ordre donné par l’utilisateur est celui qui va être réalisé.
- On envoie dans la variable pinMoteurD la valeur analogique 200 qui correspond à l’actionnement du moteur de droite avec une valeur de 200 (la valeur maximal est de 255 mais on ne prend que 200 car la voiture à tendance à patiner pour une valeur analogique de 255, on y reviendra dans la partie VI)
- On envoie dans la variable pinMoteurG la valeur analogique 200 également.

Et maintenant que se passe-t-il concrètement ? Et bien la voiture avance tout simplement !
Pour tourner à gauche les valeurs analogiques sont 200 pour le moteur de droite et 0 pour le moteur de gauche. En effet nous n’avons pas d’axes de directions dans notre voiture donc le seul moyen de tourner est de faire tourner un seul des deux moteurs.
Pour tourner à droite les valeurs analogiques sont 0 pour le moteur de droite et 200 pour le moteur de gauche.

Voici les programmes pour tourner à gauche et à droite :

Programme pour tourner à gauche :

```
if (ordre =="gauche"){
  Serial.println(ordre);
  Serial.println("La voiture tourne a droite");
  analogWrite(pinMoteurD, 255);
  analogWrite(pinMoteurG, 0);

}
```

Programme pour tourner à droite :

```
if (ordre =="droite"){
  Serial.println(ordre);
  Serial.println("La voiture tourne a droite");
  analogWrite(pinMoteurD, 0);
  analogWrite(pinMoteurG, 255);
  }
```

Enfin nous avons également ajouté l'ordre pour arrêter la voiture. L'utilisateur doit simplement dire "stop" et la voiture s'arrête.

```
if (ordre =="stop"){
    Serial.println(ordre);
    Serial.println("La voiture s'arrete");
    analogWrite(pinMoteurD, 0);
    analogWrite(pinMoteurG, 0);
    }
```

Le seul problème que nous avons est pour la marche arrière. Nous y reviendrons plus en profondeur dans la prochaine partie.

<u>Test avec la voiture :</u>

Une fois le programme compilé dans le microprocesseur du véhicule, les tests sont pas concluant. Les ordres "avancer", "gauche", "droite" et

“stop” fonctionnent bien et la voiture réagit aux ordres. Cependant, on observe un manque d’adhérence entre le véhicule et le sol. Pour cela nous avons baissé la puissance des moteurs, la valeur analogique envoyé aux moteurs est 200 alors que la valeur max est de 255. Nous utilisons donc seulement 78% de la puissance maximum que l’on pourrait utiliser. Cela réduit sensiblement le problème mais pas lorsqu’on souhaite tourner à gauche ou à droite. On peut pour résoudre ce problème ajouter du poids sur la voiture ce qui permet de pallier à ce problème. Mais nous allons développer les améliorations possibles ainsi que les problèmes que nous avons rencontrés dans ce projet plus en détails dans la prochaine partie.

Conclusion sur la partie IV :

Notre prototype fonctionne. Il répond aux ordres qui lui sont donnés. Cependant, nous avons du faire de nombreux choix. Tout d’abord la voiture manquant d’adhérence, on a du baisser la puissance des moteurs et rajouter du poids ce qui réduit la performance de la voiture. De plus pour tourner à gauche et à droite, la voiture ne possédant pas d’axe de direction (c’est à dire que les roues ne tournent pas), pour réaliser les actions “droite” et “gauche” nous avons décidés de faire fonctionner uniquement un des deux moteurs seulement. Cela réduit encore plus la performance de la voiture. Nous pouvons donc encore largement améliorer notre système malgré les difficultés que nous avons rencontrés dans notre projet et c’est ce que nous allons voir dans la prochaine partie.

## Partie V : Difficultés rencontrées et améliorations possibles :

### Difficultés rencontrées :

Il a été mis à notre disposition une voiture programmable sur Arduino. Tout d'abord, il a fallu modifier cette voiture pour qu'elle puisse répondre aux actions demandées, qui seront, tourner à droite, tourner à gauche, avancer et reculer.
Un des pignons dans un des réducteur était cassé, nous avons donc pris la décision de passer le véhicule en 2 roues motrices (en traction). Pour cela il nous a suffit d'enlever un engrenage présent dans le réducteur pour rendre la roue "libre" (elle n'était plus reliée à un axe de direction).
Un nouveau câblage à été nécessaire suite à la suppression des 2 moteurs. Les moteurs de la roue avant droite et de la roue avant gauche sont contrôlés séparément et sont branchés sur les sorties M1 et M2 (voir schéma dans la partie précédente).
Comme on l'a vu précédemment, la voiture a un réel manque d'adhérence avec le sol si l'on ne rajoute pas de poids ou si l'on ne réduit pas la puissance des moteurs. Il faut donc faire un compromis entre la vitesse de la voiture (forcément réduite par le fait qu'on réduise la puissance des moteurs) et l'adhérence.
Pour la marche arrière ou l'action de reculer, nous nous sommes confrontés également à des problèmes, Pour un moteur à courant continu, on ne peut pas envoyer de courant alternatif (donc potentiellement négatif) dans le moteur pour le faire tourner dans l'autre sens et ainsi faire reculer la voiture. En effet pour un moteur de ce type il est nécessaire d'utiliser un pont en H ou une MLI (Modulation de Largeur d'Impulsion) pour inverser la tension aux bornes du moteur. On peut également utiliser un shield moteur particulier qui permettrait aux moteurs de tourner dans l'autre sens.

### Améliorations possibles :

Notre projet permet à un utilisateur de contrôler un drone à l'aide de sa voix. Mais le projet n'est encore qu'à un stade de prototype. Nous avons plusieurs idées pour améliorer notre système. Tout d'abord comme il l'a été dit précédemment, nous n'avons pas de marche arrière sur notre voiture mais nous savons qu'il est possible d'en ajouter une. Il faut pour cela ajouter un pont en H ou une MLI ou un shield moteur afin d'ajouter cette fonctionnalité.
Nous pourrions également tenter de rajouter deux autres moteurs sur les roues arrières pour avoir 4 roues motrices. Ainsi on pourrait bénéficier

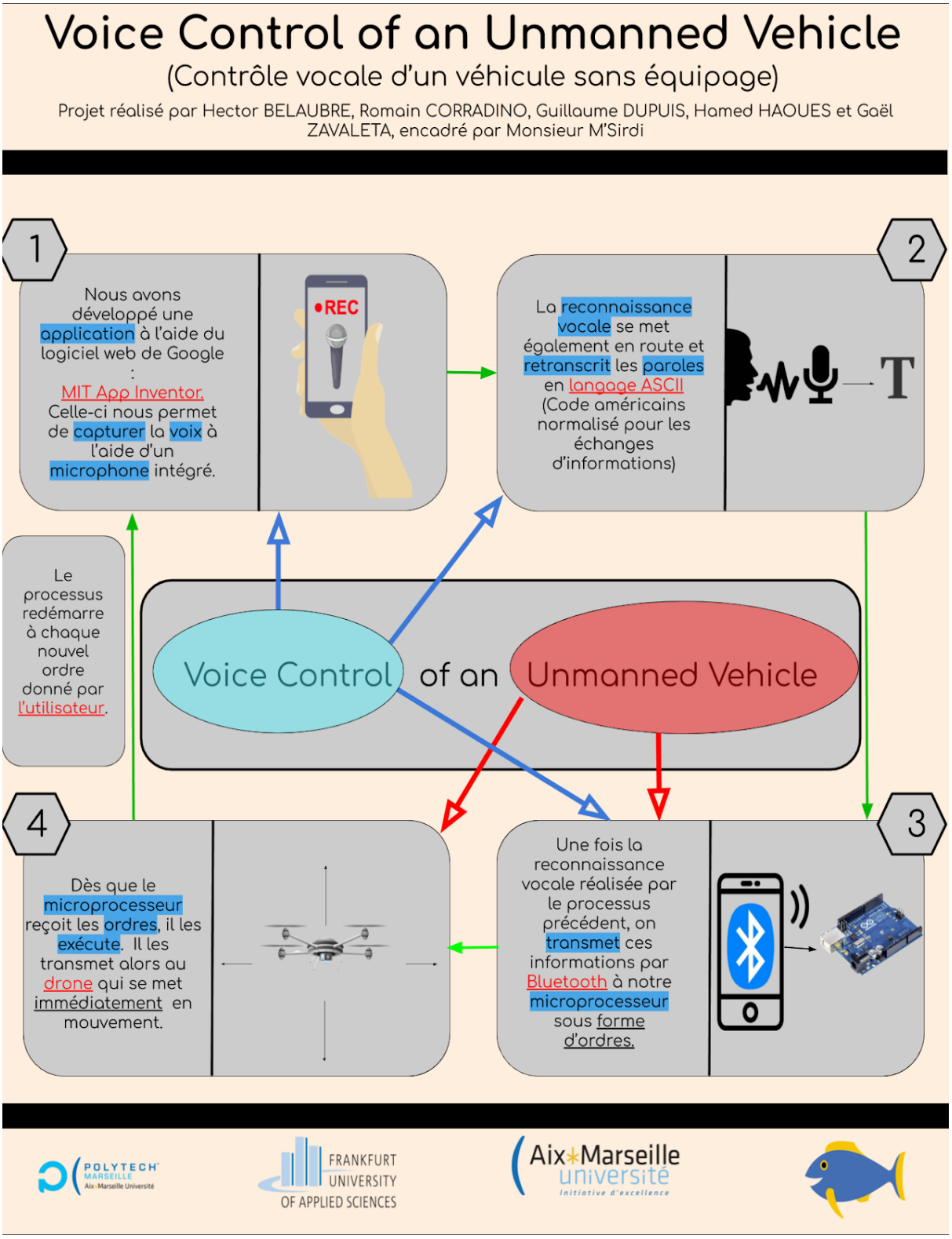
d’une meilleure adhérence et d’une meilleure traction pour le drone. Une autre modification que nous pourrions ajouter est un train avant mobile afin de pouvoir tourner comme une voiture classique, sans avoir à stopper certains moteurs.
Nous voudrions également ajouter un capteur à ultrasons. En effet ce type de capteur envoie un signal analogique sous forme de nombre au microprocesseur. Plus un objet détecté par le capteur est éloigné, et plus la valeur renvoyé est élevé. Inversement, plus l’objet détecté est près, et plus la valeur renvoyé est faible. On peut donc décider que si le capteur à ultrasons détecte des objets à une distance inférieur à un certain seuil alors le véhicule s’arrête automatiquement. Cela permettrait à la voiture de ne pas recevoir de chocs.
Nous pourrions également essayé d’améliorer l’application. En effet bien que très fonctionnels, la reconnaissance met un peu de temps à se réaliser. La transmission Bluetooth est quant à elle très rapide (la télécommande manuel intégrée envoie les ordres quasi instantanément et sont directement réalisé . Pour réduire la reconnaissance vocale, nous pourrions par exemple tenté de développer une nouvelle application mais en passant par Android SDK. Ce logiciel est bien moins facile à apprendre que MIT App Inventor mais peut-être que la reconnaissance vocale serait plus rapide.
Nous avons également réfléchi à asservir notre système. Par exemple un asservissement pourrait permettre, à l’aide de capteurs angulaires, de faire tourner le drone d’un angle de 45° exactement. Ou bien de 64° aussi. Ajouter un asservissement permettrait à l’utilisateur de contrôler bien mieux la voiture. Mais cela implique d’ajouter un train avant mobile sur un axe au préalable.
Enfin nous pourrions envisager d’adapter notre système terrien à un système aérien. Passer donc de notre voiture à un drone volant. Cela impliquerait sûrement de nouvelles problématiques mais en supprimerait d’autres comme le manque d’adhérence par exemple.

Conclusion sur la partie V :

Bien que notre prototype fonctionne, il reste beaucoup de travail pour avoir un système sophistiqué et qui répondrait à un cahier des charges concret. Une des priorités si dans le futur nous devions continuer notre projet serait la mise en place d’une marche arrière ainsi que d’une meilleur direction afin d’envisager un potentiel asservissement.

Partie VI : Conclusion et poster récapitulatif de notre projet

Voici un poster récapitulatif du fonctionnement de notre projet étape par étape :



Conclusion :

Ce projet aura été très instructif, que ce soit sur le plan humain ou sur le plan du travail.

En effet, ce projet fut une très bonne expérience pour nous préparer à notre future carrière d'ingénieur. Comme nous le savons, nous allons être confrontés pendant tout le reste de notre vie à l'élaboration de projets en tous genres.

Notre projet fut un très bon exercice préparateur à ce sujet, car nous avons dû dans un premier temps faire des recherches sur les drones, pour ensuite chercher des solutions à notre problème qui était de contrôler un drone avec la voix. Nous avons ensuite finalement enchainés par la création d'un prototype, symbolisant l'aboutissement de notre projet.

Avec ce dernier, nous avons également pu expérimenter un vrai travail d'équipe en répartissant les différentes tâches afin de travailler dans de bonnes conditions. Cela a aussi pour but de nous préparer à travailler en équipe dans un futur proche, en tant que futurs ingénieurs.

Ce travail fut également très intéressant technologiquement. En effet, les robots et autres drones représentent l'avenir de notre civilisation. On peut voir de plus en plus d'objets connectés devenir autonomes chaque jour, en passant de la normalisation des voitures autonomes avec Tesla à la fabrication de nombreux drones multifonctions par Amazon. Ces derniers participent à la créations de drones pouvant livrer des colis, ou encore à des drones pouvant aider les forces de l'ordre dans leur métier, et étant également contrôlés par la voix.

Le futur promet donc d'être radieux pour les drones et autres objets connectés. la question qu'on peut se poser est jusqu'où l'expansion du phénomène du drone ira-t'elle, et à quel point cette dernière influera-t-elle notre quotidien ?

Sources :


- www.domotique-info.fr
- www.seeedstudio.com
- www.aero-hesbaye.be
- https://fr.wikipedia.org
- www.futura-sciences.com
- www.mathworks.com
- www.atollic.com
- https://www.science-et-vie.com/archives/la-revolution-drones-13290
- https://ins2i.cnrs.fr/fr/cnrsinfo/comment-securiser-lutilisation-des-drones
- https://www.futura-sciences.com/tech/actualites/robotique-apres-drones-amazon-teste-robot-livraison-74772/
- http://www.mon-club-elec.fr
- https://www.federation-drone.org/
- https://www.arduino.cc/
- http://fritzing.org/
- http://appinventor.mit.edu/
- https://www.aranacorp.com/fr/arduino-et-le-module-bluetooth-hc-06/
- https://openclassrooms.com