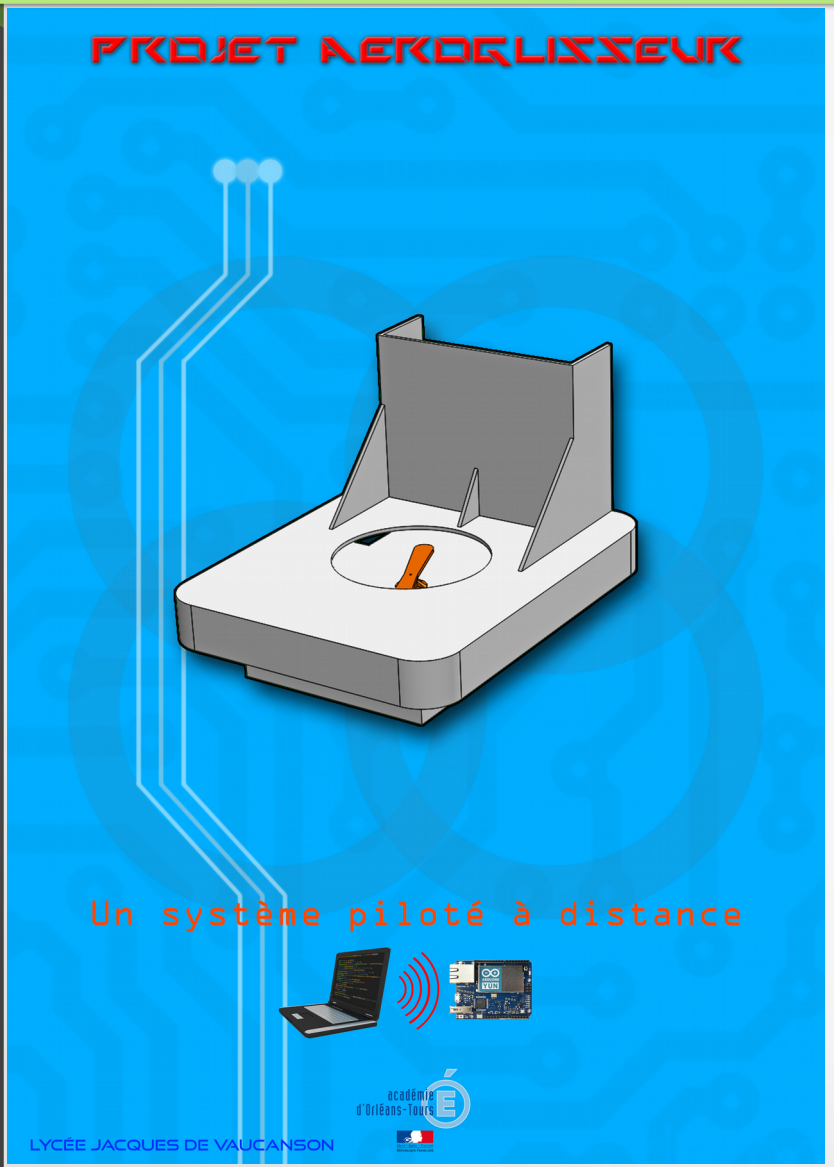
Projet aéroglisseur



Ce projet a pour but de concevoir un aéroglisseur de taille modélisme répondant à des critères :

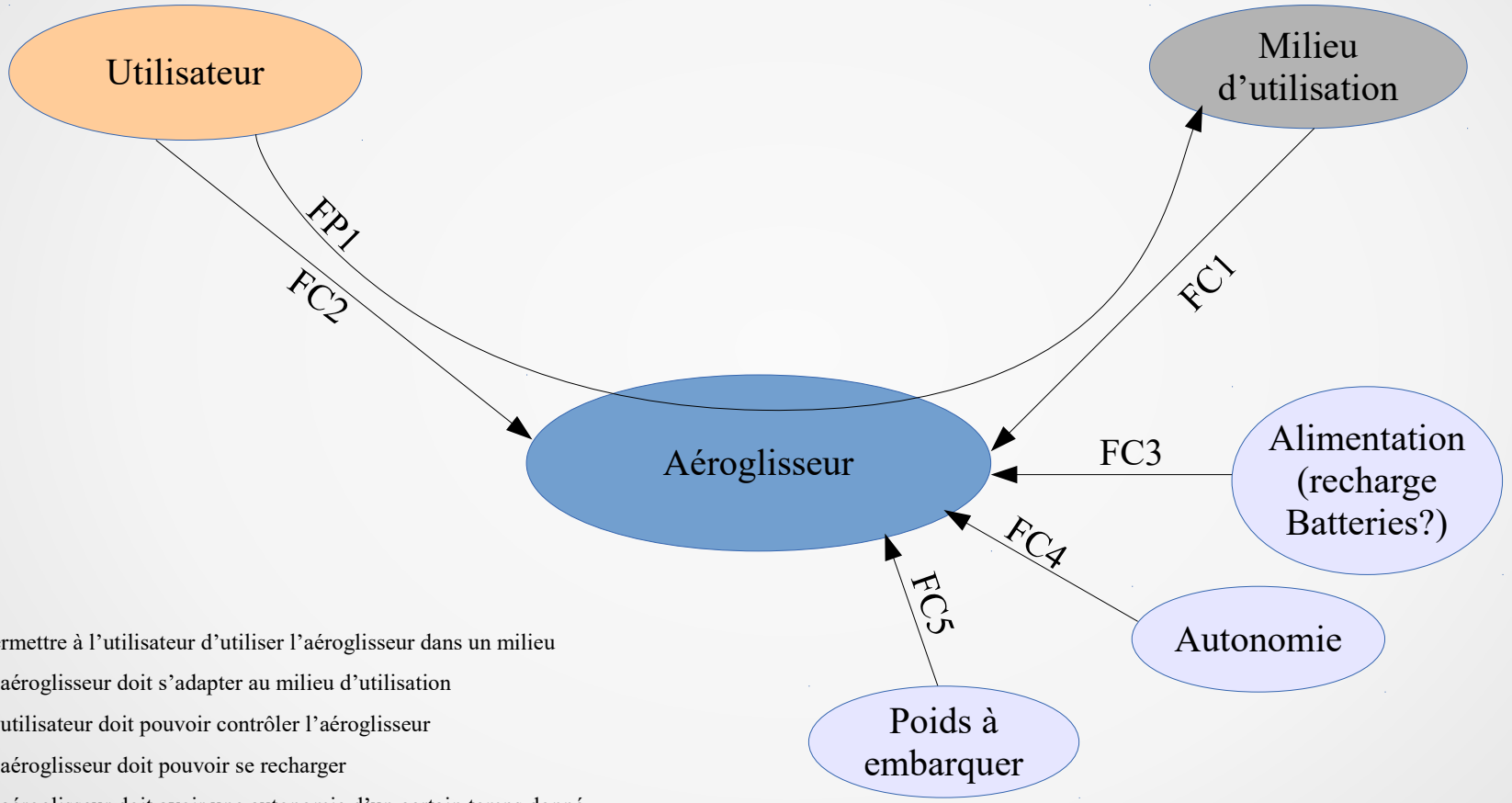
- se déplacer facilement dans un environnement respectant des conditions
- être commandé à distance
- donner une vision de son environnement proche à l'utilisateur.

COUCHOUD Thomas

Plan

- **Présentation des besoins**
- **Répartition des tâches**
- **Partie électronique – Transmission des données**
  - Acquisition
  - Transmission
  - Réception
  - Interprétation
  - Expérimentations
- **Partie visualisation**
- **Conclusion**

Les besoins



FP1 : Permettre à l'utilisateur d'utiliser l'aéroglisseur dans un milieu

FC1 : L'aéroglisseur doit s'adapter au milieu d'utilisation

FC2 : L'utilisateur doit pouvoir contrôler l'aéroglisseur

FC3 : L'aéroglisseur doit pouvoir se recharger

FC4 : L'aéroglisseur doit avoir une autonomie d'un certain temps donné

FC5 : L'aéroglisseur doit pouvoir embarquer une masse donné

Répartition des tâches

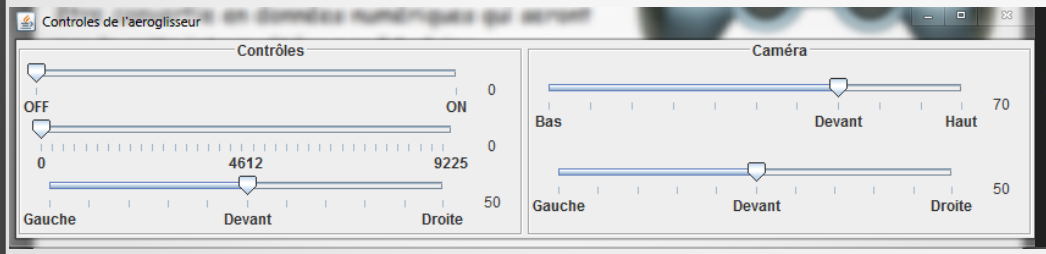
CHALUMEAU Pierre	Designer l'aéroglisseur en intégrant les solutions et en répartissant les masses
COUCHOUD Thomas & JACQUES Johann	Rendre l'aéroglisseur autonome et contrôlable à distance
DUTREUIL Raphaël	Réaliser la sustentation (élever et stabiliser l'aéroglisseur)
FROGER Olivier	Trouver un moyen permettant de diriger et propulser l'aéroglisseur

Récupérer les informations

Comment récupérer les commandes de l'utilisateur ?

Cette partie propose de numériser les demandes de l'utilisateur. Pour des raisons d'ergonomie, nous avons choisi un contrôleur de jeu USB.

Chaque touche sera reliée à une action qui pourra être convertie en données numériques qui seront pas la suite interprétées par l'Arduino.




```
 * Called when a button is pressed.
private void onButtonPressed(final String name)
{
    Main.logger.log(Level.INFO, "Button " + name + " pressed");
    if(name.equals(ACT_SUSTENT))
        Interface.changeValue("st", -9999);
}
```

Récupérer les informations

```
 * Called when a button is pressed
private void onButtonPressed(final String name)

 * Called when a button is released
private void onButtonReleased(final String name)

 * Called when an axis value changed
private void onAxisValueChange(final String name, final float value, boolean newValue)

 * Called when a POV axis value changed
private void onPovValueChange(final String name, final float value)
```

← Réception de l'événement

```
private void onButtonPressed(final String name)
{
    Main.logger.log(Level.INFO, "Button " + name
    if(name.equals(ACT_SUSTENT))
        Interface.changeValue("st", -9999);
}
```

← Traitement de l'événement

```
synchronized public static void addToSend(final String key, final int value)
{
    Main.logger.log(Level.FINEST, "Changing " + key + " to " + value);
    requests.put(key, value);
}
```

← Stockage si nécessaire

Transmission des informations

Comment permettre à l'utilisateur de communiquer avec l'aéroglisseur ?

Cette fonction est assurée par une communication WIFI entre le PC de l'utilisateur et l'Arduino contrôlant l'aéroglisseur.
Le PC va tout simplement se connecter sur une adresse internet gérée par l'aéroglisseur en donnant deux paramètres :

- Un clef identifiant le paramètre à changer (vitesse, direction...)
- Une valeur quantifiant ce paramètre

Transmission des informations

```
synchronized private String sendGet(String key, String value) throws Exception
{
    Date startDate = new Date();
    URL url = new URL(IP_URL + "arduino/" + key + "/" + value);
    HttpURLConnection connection = (HttpURLConnection) url.openConnection();
```

Format d'envoi des données :

http://<IP_ARDUINO>/arduino/<CLEF>/<VALEUR>

Exemple pour allumer la sustentation :

http://<IP_ARDUINO>/arduino/st/1

Réception des informations

```
YunClient client = server.accept();
if (client)
{
  String command = client.readString();
  command.trim();
  printMessage("Command recevied : " + command);
  String result = decrypt(command);
  printMessage("Result : " + result);
  client.print(result);
  client.stop();
}
```

← Réception de la commande

```
return writeToMotors(input.substring(0, input.indexOf('/')), input.substring(input.indexOf('/') + 1).toInt());
```

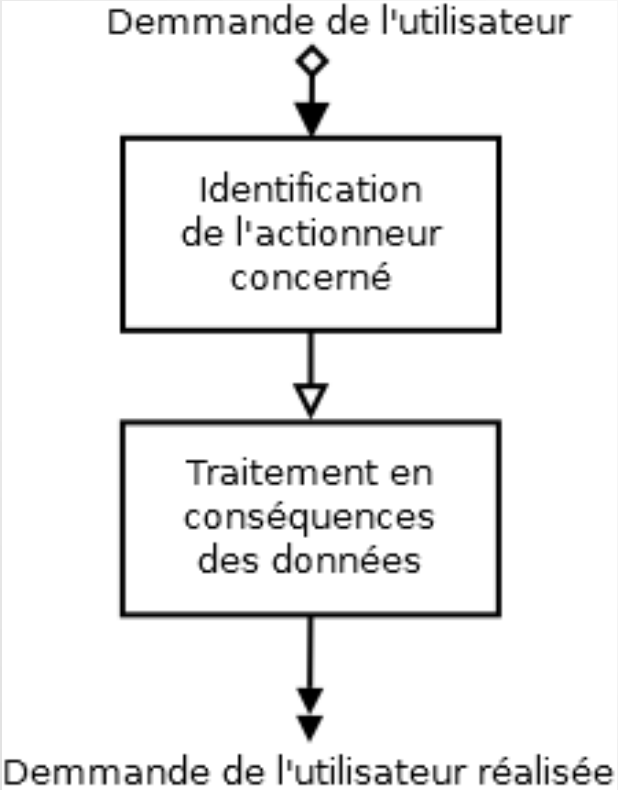
↑

Découpage de la clef et valeur

Découpage : st/1

Contrôler les composants

Comment gérer les données reçu par la carte Arduino et les adapter aux besoins de l'aéroglisseur ?



- Pour répondre à cette question on procède en 4 étapes :
  - Récupérer les informations
  - Identifier l'actionneur concerné
  - Convertir les données numériques en valeur comprises par celui-ci
  - Lui envoyer des données converties

```
printMessage("Receiving key " + key + " with value " + value);
if(key == speed_key)
{
    motorDirection.write(RPMToServoSpeed(value));
```

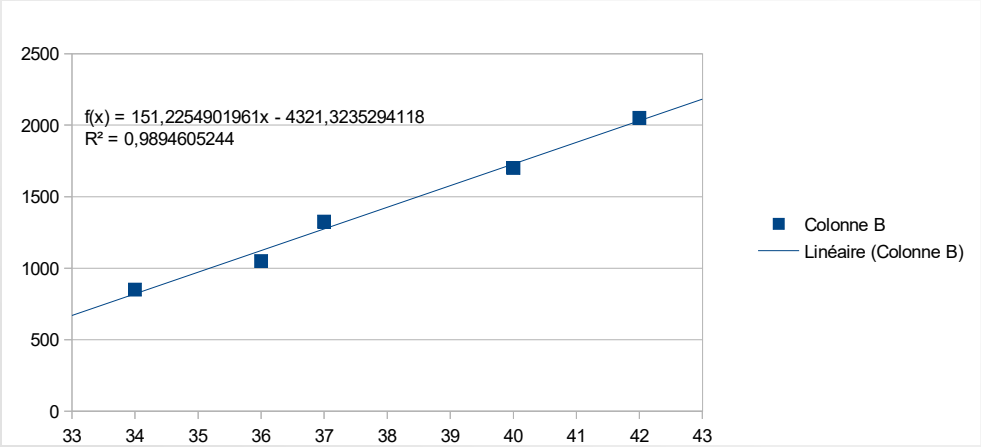
Expérimentations

Afin de tester le traitement des informations et établir une fonction de convertissage, il est nécessaire d'élaborer des représentations de l'effet des variables sur le système considéré.

Dans notre cas, ceci s'applique pour la vitesse de rotation des hélices ainsi que l'orientation des servo-moteurs.

Le protocole pour réaliser ceci:

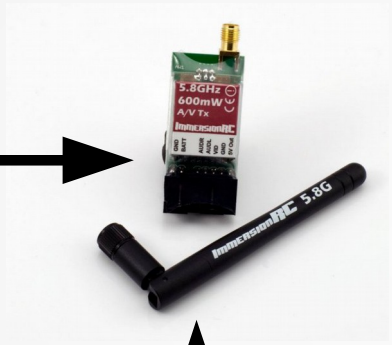
- Mesure de la vitesse (ou orientation) pour différentes valeurs données.
- Création d'un graphique des valeurs mesurées en fonction des valeurs envoyées
- Interprétation de ce graphique (souvent une droite)
- Création d'une fonction permettant le passage de la valeur reçue à la valeur « réelle »




```
return (119 + (int)(rpm / 151.22));
```

Visualisation

Connexion filaire



Acquisition de l'image

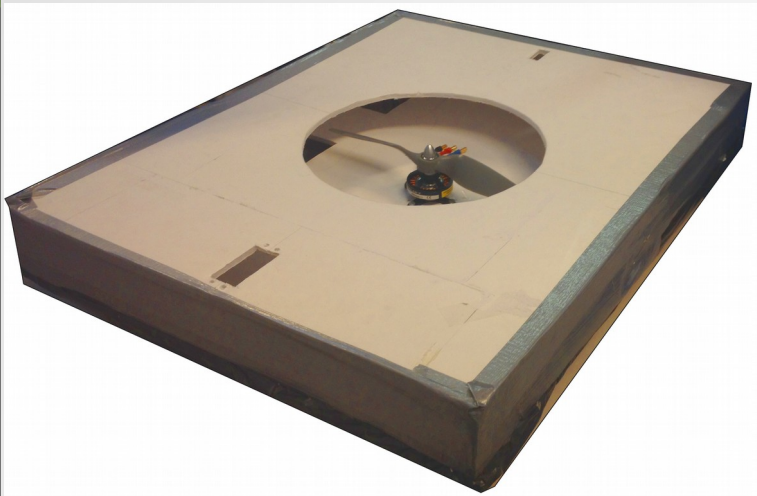
Envoi des données

Réception de l'image

Fonctionnement à 5,8GHz

- Dégradation de l'image notable : ~40m
- Réception de l'image non effectuée : ~45m

Conclusion



A l'aboutissement de notre projet l'aéroglisseur est opérationnel, aussi bien structurellement que électroniquement. En réponse aux objectifs imposés au projet, il est contrôlable à distance et capable de visionner son environnement proche.

D'un point de vue critique nous estimons que nous aurions pu pour optimiser le temps de réalisation :

- Se baser sur un autre système de communication que le wi-fi